DESIGN METHODOLOGIES FOR SCALABLE AND RELIABLE MEMORY SYSTEMS

A Dissertation
Presented to
The Academic Faculty

by

Dae Hyun Kim

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
August 2017

DESIGN METHODOLOGIES FOR SCALABLE AND RELIABLE MEMORY SYSTEMS

Approved by:

Dr. Linda Milor, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Sudhakar Yalamanchili
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Abhijit Chatterjee
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Haomin Zhou
School of Mathematics
Georgia Institute of Technology

Dr. Madhavan Swaminathan
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: April 20, 2017

To my family

ACKNOWLEDGEMENTS

I would like to begin by expressing my sincere gratitude to my research advisor, Dr. Linda Milor. She has provided me with great opportunities and autonomy to choose research topics and conduct independent research. With her insightful comments, technical creativity, and high standards of quality of research, presentations, and publications, she inspired me to explore novel technical issues and encouraged me to complete my research. Without her support and guidance, I could not have completed my doctorate.

I would like to give special thanks to Dr. Abhijit Chatterjee and Dr. Madhavan Swaminathan for serving on my thesis reading committee. I also express my appreciation for their invaluable comments and ingenious suggestions. I would like to extend my gratitude and appreciation to the other committee members, Dr. Sudhakar Yalamanchili and Dr. Haomin Zhou, for their constructive feedback and discussion.

I have enjoyed my life with friends and colleagues at Georgia Tech. I had great pleasure of working with my labmates: Woongrae Kim, Soonyoung Cha, Kexin Yang, Taizhi Liu, Rui Zhang, and Shu-Han Hsu. I am also grateful to CARET members, Prashant Nair, Jian Huang, Chia-Chen Chou, and Dr. Moinuddin K. Qureshi, for their insightful discussions and helpful comments in the area of computer architecture. I also thank Insik Yoon, Paul K. Jo, Taigon Song, Duckhwan Kim, Taesik Na, Jonghwan Ko, Kwanyeob Chae, and Jaewoong Sim for their help and support during my time at Georgia Tech. I would like to express my special thanks to Donghyuk Lee, Woojin Lee, and Seung-Jun Bae for their emotional support and great guidance.

My family has always supported me with relentless love and encouragement. To my parents, Yongsoo Kim and Myungsoon Ko and my parents-in-law, Joonsoo Kim and Gilsoon Kim, I would also like to extend my deepest gratitude for their unconditional love and profound belief in my abilities. Finally, I would like to express my deepest love and gratitude to my beloved wife, Hyunmi Kim, and my amazing daughters, Christine Yein Kim and Rebecca Yedam Kim. The completion of my dissertation would not have been possible without all their endless love, encouragement, and devotion.

Dae Hyun Kim
Atlanta, Georgia, April, 2017

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

BTI	Bias temperature instability
HCI	Hot-carrier injection
TDDB	Dielectric time-dependent breakdown
GTDDB	Gate dielectric time-dependent breakdown
MTDDB	Middle-of-the-line time-dependent breakdown
BTDDB	Backend time-dependent breakdown
SIV	Stress-induced voiding
EM	Electro-migration
NBTI	Negative bias temperature instability
PBTI	Positive bias temperature instability
BIST	Built-in self-test
BIRA	Built-in repair analysis
BISR	Built-in self-repair
DIMMs	Dual in-line memory modules
SECDED	Single-error correcting and double-error detecting
ECCs	Error-correcting codes
FIT	Failures in time
PPR	Post-package repair
SEU	Single event upset
SRAM	Static random access memory
DRAM	Dynamic random access memory
VRT	Variable retention time

SCE	Short channel effect
3D	Three-dimensional
GIDL	Gate-induced drain leakage
WL	Wordline
AVERT	An elaborate model for simulating VariablE Retention Time in DRAMs
RTN	Random telegraph noise
TA-GIDL	Trap-assisted gate-induced drain leakage
TA-GL	Trap-assisted gate leakage
ppm	Parts per million
PDK	Process design kit
1T-1C	One transistor and one capacitor
6T	Six-transistor
3T	Three-transistor
eDRAM	Embedded dynamic random access memory
WR	Write
RD	Read
CAS	Column address strobe
Emerald	Estimating MEmory ReliAbiLity Degradation
tRET	Retention time
tWR	Write recovery time
tRP	Precharge time
CRA	Counter-based row activation
PRA	Probabilistic row activation
RAC	Row activation counter
ATE	Automatic test equipment

LLC	Last-level cache
AR	Address reconfiguration
FD	Fault detection
FI	Fault identification
FR	Fault repair
CE	Correctable error
UE	Uncorrectable error
HFD	Hard-fault detection method
FALT	Fail-address lookup table
LRU	Least-recently used
FIFO	First-in first-out
LFU	Least-frequently used
MRS	Mode register set
DDR3	Double data-rate type three
SDRAM	Synchronous dynamic random access memory
CSL	Column-selection lines
XOR	Exclusive-or
RA	Redundancy algorithm
DECTED	Double-error correction and ternary-error detection
BL	Bitline
SN	Storage node
ALT	Accelerated life test
SEC	Single-bit error correction
DEC	Double-bit error correction
DOE	Design of experiments

SUMMARY

The objective of the thesis is to develop design methodologies for scalable and reliable memory systems in the presence of scalability and reliability issues exacerbated or created by continuous scaling. In this research, after investigating the origins and device-level models of memory failures caused by variable retention time, row hammering, and wearout, to examine the impact of such failures on operations of a memory system, this dissertation proposes circuit- and system-level modeling and simulation methods. With significant observations from simulation results, this dissertation introduces design methodologies that mitigate row-hammering phenomenon by employing counter-based or probabilistic row activations and repair increasing wearout failures by exploiting error-correcting codes for the error detection and sequence of commands for error identification during field operations. To enhance the reliability of a memory system, this dissertation proposes accurate memory reliability estimation and diagnosis methodologies using a system-level accelerated life test with a built-in self-test and error-correcting codes. This dissertation also introduces a method of optimizing the design of experiments that isolates a failure caused by a target wearout mechanism from failures caused by other mechanisms and minimizes errors in the estimation of wearout parameters at the normal operating condition. In conclusion, this dissertation supports continuous scaling and improves memory reliability by proposing modeling, simulating, mitigating, and characterizing schemes of reliability degradation in a memory system resulting from failures caused by variable retention time, row hammering, and wearout.

CHAPTER 1. INTRODUCTION

1.1 Problem Statement

Advances of fabrication process technology toward small feature sizes have enabled modern VLSI systems with low power, high performance, and small dimensions. However, while maintaining the pace of the doubling of the number of transistors per square inch on integrated circuits every two years, further shrinking feature sizes of transistors below 14 nm is no longer cost effective for production. Therefore, Moore's law has slowed down. In addition to *scaling wall*, which we are expected to confront in the near future, such small feature sizes of transistors not only exacerbate several failure modes but also create new failure modes in computer systems.

1.2 Prior Work

Among such reliability concerns, wearout is prevalent in all fabrication processes, front-end/middle/back-end of the line. Frontend wearout results from bias temperature instability (BTI), hot-carrier injection (HCI), and gate dielectric time-dependent breakdown (GTDDB). As a device ages, time-dependent dielectric breakdown occurs in not only the frontend but also the middle and back-end of the line, also known as middle-of-the-line/backend time-dependent breakdown (MTDDB/BTDDB). In addition to BTDDB, stress-induced voiding (SIV) and electro-migration (EM) are known as major causes of backend wearout. Prior studies have investigated mechanisms of such wearout over all processes of IC fabrication.

Prior studies [1]-[4] have observed that as a VLSI system operates over time, the threshold voltage of a device shifts resulting from BTI and HCI. BTI is generally explained using traps at the gate oxide interface and in the oxide. Since the trap density of an oxide increases as a device ages and trapped carriers change the electrical field in the oxide of a device, the threshold voltage of a device varies over time. BTI occurs in both PMOS and NMOS devices. While negative bias temperature instability (NBTI) increases in the threshold voltage of PMOS devices, positive bias temperature instability (PBTI) increases the threshold voltage of NMOS devices. HCI can also shift the threshold voltage of the CMOS transistors. Unlike BTI, which is stressed under DC bias condition, HCI is stressed when the devices operate with high switching activity. A high electrical field between a source and a drain of a MOS device generates a hot carrier, so it may drift into the oxide and occupy a defect, which becomes a trap. Such a trapped hot carrier in an oxide also changes the electric field in the oxide of a device, which shifts the threshold voltage of a device.

Several time-dependent dielectrics such as GTDDB, MTDDB, and BTDDB cause an electrical fault, or a resistive bridge fault, in VLSI systems. Using the charge trapping and detrapping model and the percolation model, prior studies [5]-[7] have explained trap-assisted tunneling from a gate to a channel of a MOS device through a gate dielectric as the origin of a GTDDB. Trapped charges in a dielectric of a MOS device can form a path from a gate to either a source or a drain. Such a path reduces the resistance between a gate to either a source or a drain, which forms a bridge between them, causing a bridging fault. Similarly, the high electrical field between adjacent metal lines causes a BTDDB, connecting both lines [8]-[10]. Recent studies [10],[11] have found that beyond 20 nm

technology node, considerable amounts of bridging faults result from an MTDDB. The electrical field between a gate and a contact from a source or a drain to a metal layer increases the trap density in a dielectric between them, causing a bridge fault similar to a BTDDB.

Prior work has investigated backend wearout mechanisms such as EM and SIV, resulting in open failures. Historically, studies have applied various approaches to examine the EM mechanism using materials, fabrication, computer-aided design, and test methodologies [13]-[15]. EM causes a high-resistive open metal interconnect when electrical current through a metal line causes the momentum of metallic ions so that interfacial voids formed during fabrication coalesce into a large void forming an open site in the metallic lattice. Since such voids usually form at the interface between two metals, EM is highly likely to occur at the interface of a via between two different metal layers. Recent studies pertaining to SIV [16]-[18] have shown that the thermal mechanical stress between metals and dielectric materials induces the directionally-biased motion of an atom. The biased motion increases in via resistance and creates a void inside a via, resulting in timing violations and functional failures in digital systems.

Previous device- and system-level studies on testing and repairing static random access memory (SRAM) [19]-[21] have investigated the impact of wearout mechanisms on memory arrays and proposed methods that detect and repair aging errors. The previous papers on wearout testing [19]-[21] have focused on the cell-level testing technique that detects BTI or a GTDDB in a single SRAM cell. Such device-level studies have not investigated critical manufacturing and testing issues such as time and cost of testing. For reducing the cost of testing and diagnosing all possible wearout in an SRAM cell, prior

work proposed system-level BIST systems and algorithms. To enable automated test and repair of SRAMs, the prior work [22]-[25] has presented the system-level built-in self-test (BIST), built-in repair analysis (BIRA), and built-in self-repair (BISR). During the manufacturing process, such test and repair systems described in [22]-[24] detect defects in memory and repair them with redundant arrays. However, since wearout occurs after manufacturing test/repair and shipping, such test and repair methodologies are inappropriate for aging errors in a memory system. Recent research on a test and diagnosis methodology of SRAM wearout errors [25] proposed a BIST system with test patterns that statistically distinguishes one wearout mechanism from the others once the types of defects, such as functional, open, or short failures, are diagnosed using BIST. All have investigated SRAM, not dynamic random access memory (DRAM). In this research, the impact of wearout on DRAMs is investigated.

Recent field studies on DRAM errors have reported several remarkable observations [26]-[28]. First, these large-scale field studies have found that contrary to common assumptions that soft errors dominate hard errors, hard errors outnumber soft errors. Second, they have also found that errors are correlated in time and space. Although single-bit errors are dominant, correctable errors (CEs) are highly likely to be followed by errors with the same address and in the same column and row. Consequently, such errors may advance to uncorrectable errors (UEs) that cause an expensive system crash resulting in the replacement of dual in-line memory modules (DIMMs) and downtime for the system. Moreover, architectural studies have shown that memory access patterns are not uniformly random. Instead, a few rows are excessively accessed [29],[30].

Such observations proclaim that solely relying on single-error correcting and double-error detecting (SECDED) error-correcting codes (ECCs) cannot sustain a memory system with aging errors in the field. This argument is supported by plotting the reliability of a 2 GB ECC-DIMM with 25,000 to 75,000 failures in time (FITs) per billion hours of operation per Mbit in a DIMM [26]. From the simulation results, after 0.97 years of operation with 75,000 FITs per Mbit in a DIMM, 50 % of DIMMs fail. Although ECCs correct a single-bit error in a word, the presence of a corrected error in a word increases the probability of a memory failure with a double-bit error in a word that ECCs cannot cover. Therefore, allowing ECCs to correct single-bit errors is not the ultimate solution for extending the lifetime and the reliability of memory.

Studies typically targeting embedded memories [23],[24], or main memories [31], have proposed combined schemes with ECCs and repair, i.e. with BIST and/or BISR. For modern embedded memories, we conventionally employ BIST and/or BISR for testing and repairing errors at the manufacturing level and ECCs for detecting and correcting soft errors in field operations. To enhance the manufacturing yield of embedded memories [23],[24], the proposed schemes have combined ECCs and BISR for repairing both hard and soft errors. Prior work [23] has proposed a method that repairs uncorrectable faulty words with redundancies and corrects correctable faulty words using ECCs. However, such correctable single-bit errors degrade the reliability of a memory system. Another study [24] has proposed a method that distinguishes a permanent fault from a soft one using both ECCs and BISR and repairs all identified hard faults. Exploiting SECDED ECCs, such schemes [23],[24] correct permanent single-bit faults whose probability of occurrence is higher than 97 % in embedded memories [32]. Since most errors are single-bit errors,

correcting manufacturing-level single-bit errors using ECCs can save area in the spare rows and columns with a reduced number of spare rows and columns used only for faulty rows and columns. However, using ECCs for repairing single-bit errors sacrifices the error-correcting capability by ECCs in field operations, degrading the reliability of embedded memories during field operations.

Because of the large area, inadequate DRAM fabrication process technology, and test coverage issues, BIST and BISR schemes have not been implemented in commercial DRAMs. Although stacked DRAM has an optional area for BIST and BISR on the logic die, we focus on standard DRAMs that do not contain either BIST or BISR. Although prior work [32] has also investigated a post-package repair (PPR) scheme for DRAMs, such a repair scheme is used for manufacturing errors that occur during the packaging processes and burn-in tests, not for aging errors. None of the schemes listed above are sufficient for mitigating aging errors in the main memory.

1.3 Thesis Statement

The objective of the proposed research is to develop design methodologies for scalable and reliable memory systems. This work presents circuit and system supports for continuous scaling by proposing a fault-tolerant memory system, mitigating various types of memory faults resulting from random telegraph noise (RTN), wordline coupling, and wearout. For reliable memory systems, this dissertation proposes methodologies that accurately estimate reliability degradation and diagnose various types of memory failures at a circuit and a system level.

1.4 Contributions of the Dissertation

State-of-the-art research on memory errors are illustrated in Figure 1. Among memory hierarchy, this work mainly focuses on cache and main memory in a memory system. Even though various types of memories can be used as cache and main memory, since SRAM and DRAM have been widely employed, this dissertation mainly investigates scalability and reliability issues in SRAM and DRAM. In such scalability and reliability issues in memory, manufacturing errors and single event upsets (SEUs) have been studied in many prior publications [22]-[24],[28],[31],[32]. Failures caused by random telegraph noise, coupling, and wearout in memory have been recently exacerbated or created. The impact of random telegraph noise in SRAM and DRAM on circuit operations has been investigated using modeling and simulation methodologies presented in prior research [34],[35]. However, since prior work [35] only considers random telegraph noise in operating current of a cell transistor, to accurately model variable retention time (VRT) phenomenon in DRAM, this dissertation investigates random telegraph noise in leakage current of a DRAM cell. S. Irobi presents deteriorated coupling faults in memory [36]. However, wordline (WL) coupling faults in DRAM, recently noticed, has not been studied. We demonstrate the prevalence of wordline coupling faults resulting from row hammering in a modern memory system, model the threshold of row hammering in DRAMs, and propose techniques that mitigate row-hammering faults in main memory systems. As discussed in the previous chapter 1.3, vast amount of research on wearout have been conducted, mainly at a device level. This work presents modeling, simulating, mitigating, and testing methodologies for such failures in memory at a circuit and a system level, merely explored in prior research.

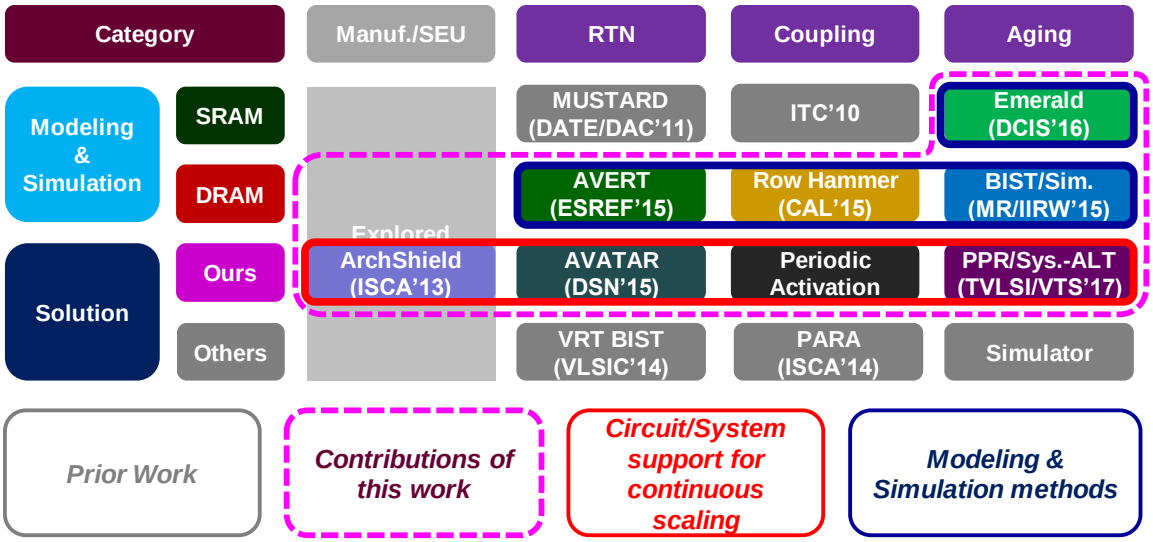


Figure 1 State-of-the-art research on failures in memory.

1.5 Thesis Organization

The remainder of the dissertation is organized as follows. Chapter 2 discusses device-level models of errors in memory caused by failure mechanisms such as variable retention time, row-hammering, and wearout, used in this dissertation. Chapter 3 presents the proposed circuit- and system-level modeling and simulating methodologies for investigating the impact of such memory errors on memory operations. Chapter 4 proposes circuit- and system-level design methodologies that mitigate such memory errors for scalable and reliable memory systems. Chapter 5 proposes system-level accelerated life test (ALT) with optimized experimental designs for accurately estimating and diagnosing circuit- and system-level memory reliability degraded by errors resulting from wearout. Finally, Chapter 6 concludes the dissertation and discusses future work.

CHAPTER 2. DEVICE-LEVEL MODELING OF ERRORS IN MEMORY

2.1 Reliability Issues in Memory

2.1.1 Random Telegraph Noise in DRAMs—Variable Retention Time

As main memory in computer systems, DRAM, which consists of one transistor and one capacitor (1T-1C), has been widely employed. In the form of charge, a DRAM cell stores data on a capacitor. A DRAM cell loses data resulting from leakage from a storage node without periodic refreshes, which a memory controller operates based on the length of time that a DRAM cell can retain data, referred to as the retention time. To meet the standard refresh rate, DRAM manufacturers require accurate characterization of the DRAM retention time. However, precisely profiling the retention time becomes a challenge in the presence of random fluctuations in retention time, also known as variable retention time. Since DRAM retention time randomly varies every test, screening all retention errors resulting from VRT during testing is prohibitive. Therefore, to enhance DRAM reliability degraded by VRT, we require a method that accurately predicts the impact of VRT on DRAM failures with the assistance of circuit simulations.

2.1.2 Wordline Coupling Errors in DRAMs—Row Hammering

For higher capacity, DRAM has continued to scale towards high-density chips with smaller feature sizes. Unfortunately, below the feature size of 45 nm, transistors suffer from short channel effect (SCE), which lowers threshold voltage, increases leakage, and reduces the retention time of DRAM cells. To mitigate SCE and maintain the retention time, DRAM vendors have exploited three-dimensional (3D) structures of cell transistors [37]-

[39]. Although such 3D cell transistors alleviate SCE, DRAM cells lose data caused by severe activations of adjacent rows, potentially resulting in data errors of DRAM cells connected to neighboring rows. Furthermore, since DRAM has scaled down to a smaller feature size and transitioned from $6F^2$ to $4F^2$ [39], such a trend reduces the distance between DRAM cells and increases coupling from neighboring DRAM cells and their wordlines. Coupling from neighboring rows increases the voltage level of a victim row, increasing standby current. Higher standby current accelerates charge leakage from DRAM storage nodes and reduces the retention time of the cell. We refer to this phenomenon of increasing leakage in cells of adjacent rows (victim rows) by frequent activations on a given row as row hammering. Note that technology shrinking exacerbates row hammering, becoming an even more serious problem. This work proposes a method that tolerates row hammering in high-density DRAM memories. When a memory controller sends a large number of activations on a specific row in the memory and no activation on the rows neighboring to the aggressor row, row hammering causes DRAM cells that are connected to victim rows to lose data. If such victim rows would get an activation and restore the data back to its original state, row hammering results in no data loss. Figure 2 illustrates the problem of row hammering, for a given row X, where the neighboring rows are labeled X-1 and X+1. If X is accessed excessively, and X-1 and X+1 are not accessed, the data of these neighboring rows can get lost resulting from row hammering. We define the threshold for the number of activations within a refresh cycle required to cause data loss resulting from row hammering as the row-hammering threshold (RH_{th}). Since such threshold reduces as technology advances to the next generation, the row hammering problem becomes much severe for future nanoscale DRAMs. We demonstrate that for the next generation DRAMs

row hammering threshold could be in the range of several tens of thousands of row activations for a given row, a threshold that can be easily reached by current workloads. Furthermore, malicious (or memory stress) programs can easily cross such a threshold. To maintain data integrity of DRAM, we should mitigate row hammering for both typical workloads as well as for worst-case (or malicious) workloads.

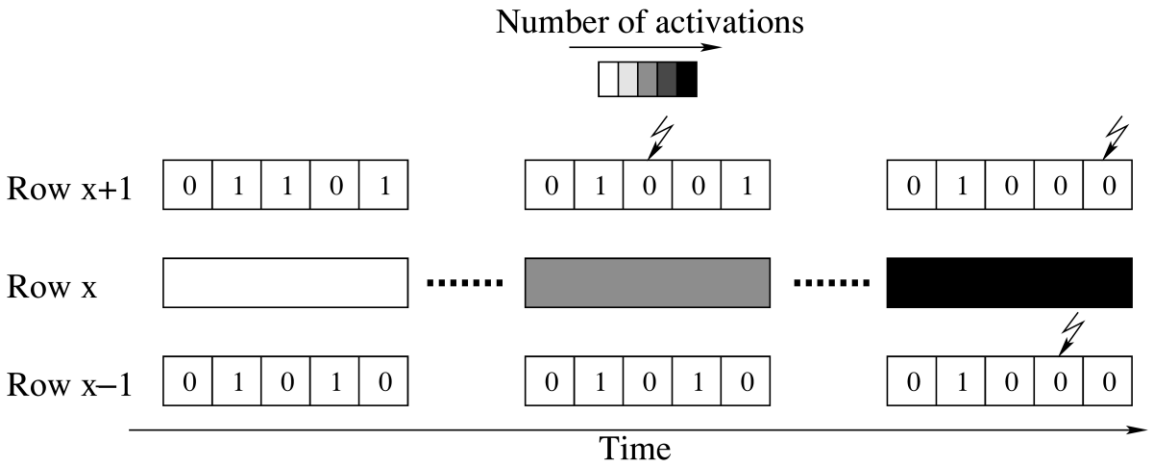


Figure 2 Impact of row hammering on neighboring DRAM cells as the number of activations on aggressor row *X* increases [30].

2.1.3 Wearout Failures in Memory

Because of continuous dimensional scaling and voltage scaling, disproportionate to dimensional scaling, high electrical fields are applied to the dielectrics of transistors and between interconnects, which degrades the reliability of computer systems. Among the wearout mechanisms, frontend wearout caused by BTI and HCI degrades the performances of transistors [40],[41]. BTI and HCI increase the threshold voltage (V_{th}) of a MOSFET device, which reduces the operating current of a device and causes performance degradation of a circuit.

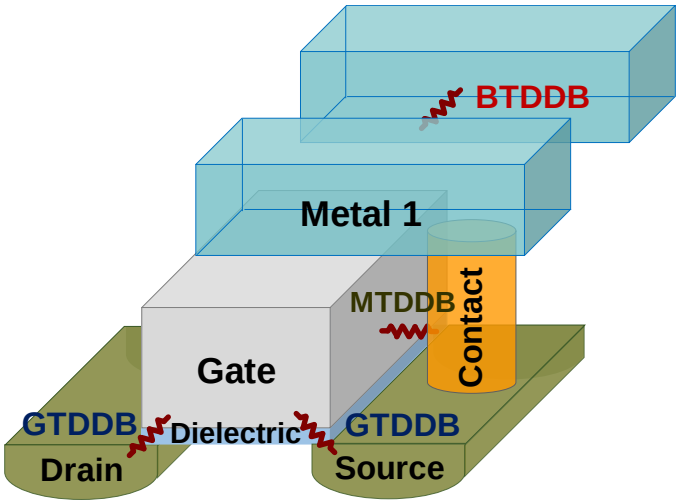


Figure 3 Definitions of the three TDDB mechanisms considered in this work: GTDDB, BTDDB, and MTDDB.

In addition to BTI and HCI, one of dominant wearout mechanisms in modern VLSI systems is time-dependent dielectric breakdown. TDDB in the front-end-of-the-line occurs in the gate dielectric of a transistor, referred to as GTDDB. In the back-end-of-the-line, TDDB between adjacent metal lines is known as BTDDB. TDDB in the middle-of-the-line has also been recently acknowledged as a significant contributor to TDDB [12],[42]-[48]. Such TDDB mechanisms in the front-end/middle/back-end-of-the-line are depicted in Figure 3. Since TDDB degrades the lifetime of a circuit/system, ensuring that the circuit/ system operates reliably throughout its specified lifetime is challenging.

2.2 Origins of Errors in Memory

2.2.1 Origins of Random Telegraph Noise in DRAMs—Variable Retention Time

DRAM retention time is inversely proportional to total leakage current [49]. A DRAM cell has various leakage sources: leakage from a storage node to a plate poly, a gate, a source (sub-threshold current), a junction, another transistor (isolation leakage), and a body (drain leakage or gate-induced drain leakage). Since gate-induced drain leakage

(GIDL) current empirically shows random telegraph noise among various current sources, trap-assisted GIDL is known as the origin of variable retention time [49]-[51]. In addition to TA-GIDL, gate leakage also experimentally exhibits RTN, also explained using traps. Therefore, trap-assisted gate leakage can be another cause of VRT [52],[53]. A trap can be randomly occupied, increasing leakage current. As a result, a cell leaks faster and exhibits a lower retention time. However, when the trap becomes empty again, leakage current reduces, resulting in a higher retention time. Such random variations in leakage current resulting from different trap conditions cause VRT in DRAMs.

2.2.1.1 Trap-Assisted Gate-Induced Drain Leakage

Figure 4(a) shows the device structure of a cell transistor and the mechanism of GIDL current. In the case of DRAM standby or precharge modes, if data '1' is written on a cell, a drain (or a storage node) is charged up to a high voltage level (V_{DD}) by a bitline (BL), and a gate (i.e., a wordline) has negative bias for reduction of the sub-threshold voltage. Since the bias between the drain and the gate is high enough to deplete the n+ drain region under the gate and cause high-field effects, such as avalanche multiplication and band-to-band tunneling, increased electrical field and band bending generate electron-hole pairs in the depletion region. While electrons that flow to the capacitor increase the GIDL current, holes that drift to the substrate contribute to an increase of substrate current [54]. Figure 4(b) shows the trap-assisted tunneling mechanisms [49]: (1) an electron moves from the valence band to a trap by thermal emission and tunnels from the trap to the conduction band; (2) an electron tunnels from the valence band to a trap and then tunnels from the trap to the conduction band like a stepping stone; and (3) an electron tunnels from the valence band to a trap, and thermal emission helps the electron move up to the conduction band from the trap.

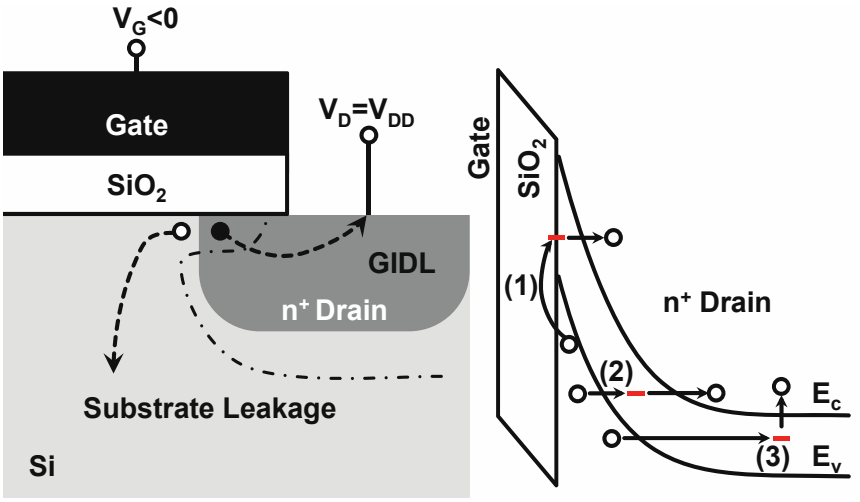


Figure 4 (a) GIDL during precharge mode in DRAMs ([54]) and (b) mechanism of trap-assisted GIDL (source [49]).

2.2.1.2 Trap-Assisted Gate Leakage

Random fluctuations in DRAM retention time result from not only trap-assisted GIDL but also trap-assisted gate leakage current [52],[53]. During a DRAM standby mode, Figure 5(a) and Figure 5(b) depict the trap-assisted tunneling and two mechanisms of gate leakage: (1) direct tunneling and (2) trap-assisted tunneling. Although the energy level of SiO2 is higher than that of the gate or the drain, the tunneling effect causes gate leakage. In other words, trap-assisted tunneling facilitates gate leakage [55],[56]. Because the number of occupied traps randomly varies, gate leakage current resulting from trap-assisted tunneling also randomly fluctuates. To explain multi-state of gate leakage fluctuation, we exploit multi-trap-assisted tunneling from recent research rather than conventional single-trap considerations [57].

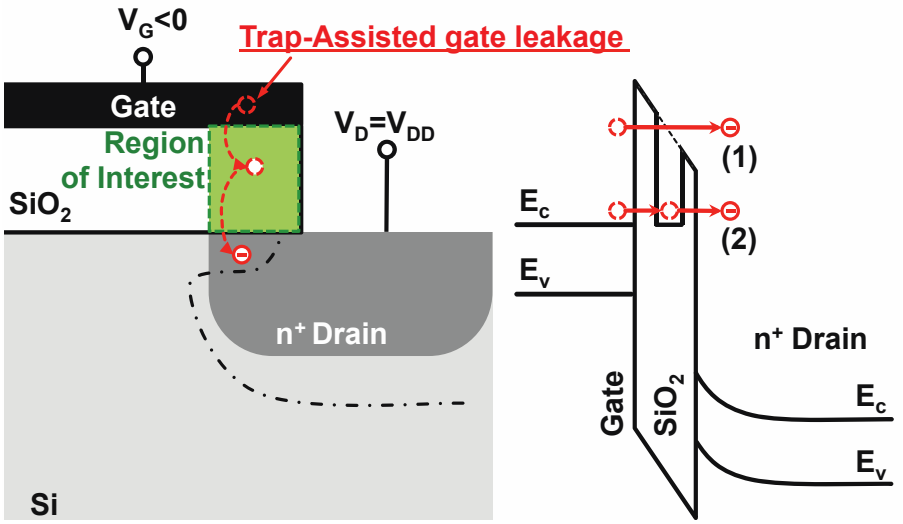


Figure 5 (a) Trap-assisted gate leakage current during a precharge mode in DRAMs and (b) mechanisms of gate leakage.

2.2.2 Origins of Row-Hammering Phenomenon

Row hammering originates from two effects: wordline to wordline coupling and passing gate effect.

2.2.2.1 Wordline to Wordline (WL-WL) Coupling

DRAM voltage does not scale down proportional to a feature size. At a smaller feature size, the ratio of coupling noise to the level of stored signal voltage on a non-accessed DRAM cell of neighboring rows increases because of WL-WL coupling [58],[59]. Coupling noise between wordlines increases the sub-threshold leakage current of cell transistors on adjacent rows [58].

2.2.2.2 Passing-Gate Effect

Although 3D transistors mitigate SCE, they are susceptible to coupling from adjacent gates and affect a victim gate [38]. A gate close to the victim gate using the same

active area is referred to as active adjacent gate. A gate close to the victim gate that do not use the same active area is called as passing gate. Activating any active adjacent gate or passing gate changes the electric field around the victim gate, which lowers the threshold voltage and increases leakage current of victim cell transistors.

2.3 Device-Level Models for Errors in Memory

2.3.1 Modeling of Random Telegraph Noise in DRAMs

2.3.1.1 Leveraging the Trapping and Detrapping (TD) Model

As illustrated in Figure 6, RTN is explained with the TD model of silicon dioxide (SiO_2) defects, which randomly capture and emit charge [60]. Since captured and emitted charge changes the energy of a trap, the variation in the leakage current of a transistor, especially TA-GIDL and TA-GL, depends on the number of captured defects. According to the TD model, the number of defects that capture and emit charge follows a Poisson distribution with a wide range of time constants for emission (τ_e) and capture (τ_c) [61]. Such constants are random variables that depend on temperature, bias, and trap location [51],[60]:

$$\tau_c = 10^p \cdot (1 + e^{-q}), \quad (1)$$

$$\tau_e = 10^p \cdot (1 + e^{q}), \quad (2)$$

where $p \in [p_{min}, p_{max}]$ and $q = (E_t - E_F)/k_B T$ [50], p is the signal frequency range on the log scale, E_F the Fermi level, k_B Boltzmann constant (eVK^{-1}), T temperature (K), and E_t the energy drop at the trap in SiO_2, which is a function of [51]:

$$E_t = \frac{T_{ox} - x_T}{T_{ox}} \cdot qV_{ox}, \quad (3)$$

$$V_{ox} = -V_{DG} - V_{FB} - \psi_S, \quad (4)$$

$$\psi_S = -V_{DG} - V_{FB} - \frac{qN_D{T_{ox}}^2\varepsilon_{si}}{{\varepsilon_{ox}}^2} + \sqrt{\left(-V_{DG} - V_{FB} - \frac{qN_D{T_{ox}}^2\varepsilon_{si}}{{\varepsilon_{ox}}^2}\right)^2 - (-V_{DG} - V_{FB})^2}, \quad (5)$$

where V_{DG} is the voltage between a drain and a gate, q is the electronic charge, ψ_S is the surface potential, V_{FB} is the flat band voltage, ε_{si} and ε_{ox} are the permittivity of the silicon and oxide, respectively, T_{ox} is the oxide thickness, x_T is the depth of the trap from the Si/SiO_2 interface, and N_D is the doping density.

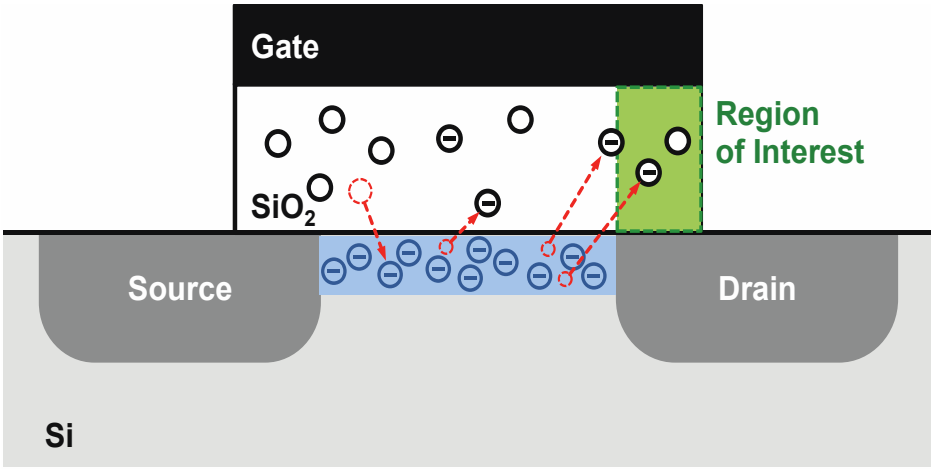


Figure 6 The charge trapping and detrapping model.

The capture and emission time constants relate to not only the drain doping concentration, oxide thickness, and the drain gate voltage, but also trap position. Different vertical positions of a trap have different trap energies, changing the ratio between the time constants of capture and emission (τ_e/τ_c), which affects the number of captured traps at any given time in a MOSFET. Therefore, for obtaining trap configurations, we need to take the vertical trap position into account. Since relevant traps are only in the overlap region between the drain and the gate, the position of a trap in the channel-length direction must be also taken into account. For every trap site in the region of interest, we calculate the probability of a trap being captured or evicted based on the Poisson distribution of a trap at a given time from equations (1) and (2). Based on the probability, charge is captured or emitted to/from each trap. We obtain the number of captured traps in a transistor at a given

time. Repeating the process generates a time-varying trap configuration within the 3D oxide structure.

2.3.1.2 Leakage current calculation

Based on the trap configuration, we calculate the variation in leakage current resulting from both GIDL current (I_{GIDL}) [51],[62] and gate leakage current (I_{gate}) [52],[63]. Regarding the TA-GIDL current, because the probability of two-step tunneling is much higher than that of the band-to-band or thermal-assisted tunneling, we assume that GIDL current by two-step tunneling is dominant [51],[62]. Therefore, GIDL current variation (ΔI_{GIDL}) can be simplified as follows [64]:

$$\Delta I_{GIDL,j} = A \cdot e^{\left(-B_{it,j}/F_j\right)}, \tag{6}$$

$$B_{it,j} = \frac{4}{\hbar} \cdot (2m_n)^{\frac{1}{2}} \cdot \frac{(E_c - E_{t,j})^{\frac{3}{2}}}{3q}, \tag{7}$$

$$\Delta I_{GIDL,tot} = \frac{A}{\Delta X} \cdot \sum_i e^{\left(-B_{it,i}/F_i\right)} \cdot \Delta x_{T,i}, \tag{8}$$

where A depends on the interface trap density, F_j is the total field in the deep depletion region, $\hbar$ is Planck's constant, m_n is the effective mass of an electron, E_c is the energy of the conduction band, Δx_T is the trap location, E_t is the energy drop at a trap in SiO_2, q is the electronic charge, and ΔX is the effective action range of electrical field. As for gate leakage, instead of the single-trap model that is less accurate in the high-stressed dielectrics [57], we utilize the multi-trap model for calculating the variation in trap-assisted gate leakage current (ΔI_{gate}) of a cell transistor. We calculate ΔI_{gate} as follows [64]:

$$\Delta I_{gate} = \left(w \cdot L_{eff}\right) \cdot q \sum_i \left(R_{c,i} + R_{e,i}\right) \cdot \Delta x_{T,i}, \tag{9}$$

where w and L_{eff} are the width and the length of the effective oxide area, q is the electronic charge, R_c and R_e are the capture and emission rate in a trap, respectively, and Δx_T is the trap location. We calculate R_c and R_e using equations:

$$R_{c,j} = \sum_{i=1}^{j-1} {\tau_{c_{i,j}}}^{-1} N_{t_k} f_{t_i}\left(1 - f_{t_k}\right), \tag{10}$$

$$R_{e,j} = \sum_{i=1+j}^{n+1} {\tau_{e_{j,i}}}^{-1} N_{t_k} f_{t_k}\left(1 - f_{t_i}\right), \tag{11}$$

where N_{t_k} and $f_{t_i} = t_{c_i}^{-1} / \left(t_{c_i}^{-1} + t_{e_i}^{-1}\right)$ are the density and the occupancy probability of a trap, respectively.

2.3.2 Modeling and Analysis of Row Hammering Threshold

The leakage current of a cell transistor (I_{leak}) increases at lower technology nodes. DRAM vendors keep a guard-band for the retention time ($t_{ret,GB}$) that is β times refresh rate ($t_{ret,th}$) as a safety margin to conform to the JEDEC refresh standard. At this guard-band, let the leakage current be $I_{leak,GB}$. Let $t_{ret,RH}$ be the time during which a cell on a victim row suffers from row hammering and $I_{leak,RH}$ be the increased leakage current by α times under row hammering. At sub-nanometer nodes, $I_{leak,RH}$ is represented by (12):

$$I_{leak,RH} = \alpha \cdot I_{leak,GB}. \tag{12}$$

Alternatively, I_{leak} can be represented by

$$I_{leak} = \frac{Q}{t} = \frac{C \cdot V}{t} \rightarrow \alpha \cdot I_{leak} \cdot t, \tag{13}$$

in which C_s is a cell capacitance of a DRAM cell and V is the capacitor driving voltage. Using (12) and (13),

$$I_{leak,GB}.t_{ret,GB} = C_s \cdot V$$

$$= I_{leak,GB} \cdot (t_{ret,th} - t_{ret,RH}) + I_{leak,RH} \cdot t_{ret,RH}$$

$$= I_{leak,GB} \cdot (t_{ret,th} - t_{ret,RH}) + \alpha \cdot I_{leak,GB} \cdot t_{ret,RH}$$

$$\rightarrow t_{ret,GB} = t_{ret,th} + (\alpha - 1) \cdot t_{ret,RH}. \quad (14)$$

Expressing $t_{ret,GB}$ in terms of $t_{ret,th}$,

$$t_{ret,GB} = \beta \cdot t_{ret,th}. \quad (15)$$

From (14) and (15), row-hammering threshold, RH_{th}, is given by

$$RH_{th} = \frac{\beta-1}{\alpha-1} \cdot M_{max}, \quad (16)$$

where M_{max} is total possible number of activations in a refresh rate. At $t_{ret,th}$ of 64 ms, M_{max} ≈1.3 million

$$RH_{th}@64ms = \frac{\beta-1}{\alpha-1} \times 1.3M. \quad (17)$$

The main component of I_{leak} is the sub-threshold leakage current (I_{sub}) from row hammering ($I_{leak,RH}$) [38],[58],[59]:

$$I_{leak} \approx I_{sub} \propto e^{q\Delta V_{th,sub}/nkT}, \tag{18}$$

in which $\Delta V_{th,sub}$ is variation in sub-threshold voltage and n is body-effect coefficient. $\Delta V_{th,sub}$ = 50 mV to 70 mV in 50 nm DRAM with SRCAT [38] and body-effect coefficient (n) = 1.1 ~ 1.4 [65]. So α ranges from 4 to 11.7. For example, for α = 11, β = 2, RH_{th} = 130 *K*, which can be expected of current generation DRAM modules. However, α is related to fabrication process and is increasing as DRAM scales down. Thus, future nanoscale DRAM can be expected to have RH_{th} in the range of few tens of thousands. To address the problem for future technology nodes, we will assume RH_{th} as 32 *K* in our study.

2.3.3 Device-Level Models for Wearout in Memory

To estimate the lifetime of an SRAM cell degraded by GTDDB, we exploit the device-level model of the characteristic lifetime for GTDDB [66], η_{GTDDB}, as follows:

$$\eta_{GTDDB} = A_{GTDDB}\left(\frac{1}{WL}\right)^{\frac{1}{\beta_{GTDDB}}} \cdot \frac{V^{a+bT}}{\alpha_{GTDDB}} \cdot F^{\frac{1}{\beta_{GTDDB}}} \cdot e^{\left(\frac{c}{T}+\frac{d}{T^2}\right)}, \tag{19}$$

where A_{GTDDB}, a, b, c, and d are fitting parameters, α_{GTDDB} is the stress probability, β_{GTDDB} is the Weibull shape parameter, W is the width of a gate, L is the length of a gate, V is the gate voltage of a transistor, T is temperature, and F is cumulative-failure percentile at use condition.

For the estimation of the time-to-failure in an SRAM cell due to BTDDB, we employ the equation of the characteristic lifetime for BTDDB [66], η_{BTDDB}, as follows:

$$\eta_{BTDDB} = A_{BTDDB} \cdot \frac{L^{\frac{1}{\beta_{GTDDB}}}}{\alpha_{GTDDB}} \cdot e^{-\gamma E^m} \cdot e^{\frac{E_A}{kT}}, \quad (20)$$

where A_{BTDDB} and m are fitting parameters, α_{BTDDB} is the stress probability, β_{BTDDB} is the Weibull shape parameter, L_{BTDDB} is a function of the vulnerable length of the dielectric segment, γ is the field acceleration factor, k is Boltzmann's constant, E_A is the activation energy, T is temperature, and E is the corresponding electrical field, which corresponds to V/s_{BTDDB}, where V and s_{BTDDB} denote the supply voltage and the space between lines, respectively.

Similar to the characteristic lifetime degraded by BTDDB, the time-to-failure of an SRAM cell due to MTDDB is estimated based on the device-level characteristic lifetime for MTDDB [11], η_{MTDDB}, as follows:

$$\eta_{MTDDB} = A_{MTDDB} \cdot \frac{L^{\frac{1}{\beta_{MTDDB}}}}{\alpha_{MTDDB}} \cdot e^{-\gamma E^m} \cdot e^{\frac{E_A}{kT}}, \quad (21)$$

where A_{MTDDB} and m are fitting parameters, α_{MTDDB} is the stress probability, β_{MTDDB} is the Weibull shape parameter, L_{MTDDB} is a function of the vulnerable length of the spacer dielectric segment between a contact and a gate, γ is the field acceleration factor, k is Boltzmann's constant, E_A is the activation energy, T is temperature, and E is the corresponding electrical field, which corresponds to V/s_{MTDDB}, where V and s_{MTDDB} denote the supply voltage and the space between a contact and a gate, respectively.

CHAPTER 3. CIRCUIT AND SYSTEM SIMULATION MEHODOLOGY OF ERROR IN MEMORY

3.1 The Algorithm and the Circuit Simulation Methodology of Random Telegraph Noise in DRAMs—AVERT

This work proposes AVERT, an elaborate device model for simulating VRT in DRAM circuits. To understand the configuration of traps, we adopt the charge trapping and detrapping model. From trap configurations, we generate random telegraph noise in trap-assisted gate-induced drain leakage (TA-GIDL) and trap-assisted gate leakage (TA-GL), known as the origin of VRT [49]-[53]. Finally, we simulate DRAM arrays with VRT by introducing gate-level models to the netlist and generating control voltage inputs converted from RTN leakage for circuit simulation using a Perl script. Employing experimental data and our model, we propose a method for determining an appropriate test time per a DRAM cell and for reducing the total test time of DRAMs considering the VRT phenomenon.

3.1.1 The Algorithm for the Device Model in AVERT

A flowchart in Figure 7 shows the algorithm of AVERT. The pseudo-code shown in Figure 8 explains the device model for generating VRT signals in DRAMs. For generating RTN in leakage current, AVERT defines trap configurations with a 3D-structure and then initializes traps by determining if traps are filled or not, based on the times of capture/emission following exponential random distributions. After initialization, AVERT generates a time variant RTN signal for a transistor until the target time (t_{target}) with a resolution of the time step (t_s). For each trap, AVERT checks if it is filled or not

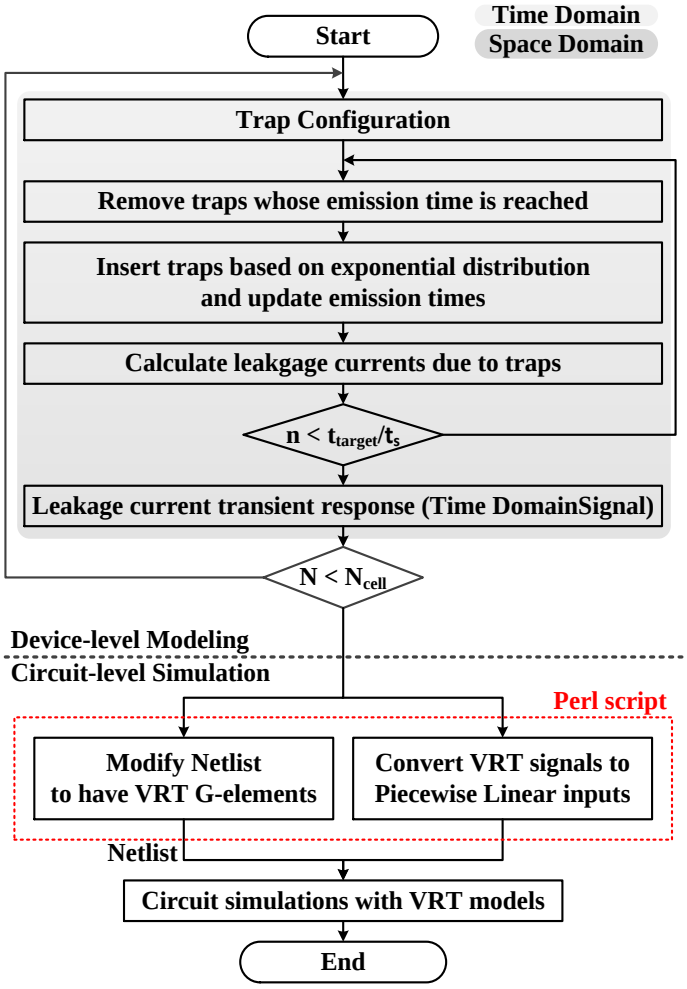


Figure 7 The flowchart of AVERT.

first. If it is filled, AVERT statistically determines whether a trap will be evicted or not based on the trap emission probability, which follows a Poisson distribution with a shape parameter of τ_e. After eviction, the capture time is updated based on the exponential random distribution. On the other hand, if a trap is not filled, AVERT determines whether a trap will be filled or not based on the trap capture probability of a Poisson distribution with τ_c. After updating the trap configuration, AVERT calculates the current variation of both gate leakage and TA GIDL. In the end, to get RTN in the leakage current of each DRAM cell in an array, AVERT repeats the generation of the time varying VRT signal for each cell with a different seed, which changes the random distribution for the trap configuration in every iteration.

Algorithm: Circuit Simulations with non-stationary RTN in AVERT

Input: Circuit simulation end time(t_{max}), length(l), width(w), thickness(t_{ox}), Mesh resolution ($N_{mesh,x}$,$N_{mesh,y}$,$N_{mesh,z}$), total number of cell transistors (N_{tr}), simulation time step (t_s)
Output: time-variant VRT signals for all cell transistors for circuit simulation [0,t_{max}]

```
//Generate different time-variant VRT signal for every cell transistor
//up to N_tr (total number of cell transistors)
n=0, x=y=z=0;
while ( n < N_tr ) do
    //Different seed for random function with different n
    seed=rand();
    //Initialization of trap configuration in a transistor
    //Define 3D device structure N_mesh (number of mesh)
    //Build the probability table of capture and emission for each mesh site
    while ( (x,y,z) < (l/N_mesh,x,w/N_mesh,y,t_ox/N_mesh,z), respectively) do
        if ( unirand(seed) < N_traps )  then
            tr(x,y,z)=1; //set trap flag as trapping
            t_c.tr(x,y,z)=exprand_t_c(y,T,V_gate); //capture time update
        else
            tr(x,y,z)=0; // set trap flag as detrapping
            t_e.tr(x,y,z)=exprand_t_e(y,T,V_gate); //emission time update
        end
        x++, y++, z++;
    end
    //Generate non-stationary leakage of a transistor up to t_max
    t=0, x=y=z=0;
    while ( t < t_max ) do
        while ( (x,y,z) <  (l/N_mesh,x, w/N_mesh,y, t_ox/N_mesh,z), respectively ) do
            if ( tr(x,y,z) == 1 & rand() < poission.prob(t_s,t_e.tr(x,y,z)) )  then
                tr(x,y,z) = 0;
                t_c.tr(x,y,z) = exprand_t_c(y,T,Vgate);
            else if ( tr(x,y,z) == 0 & rand() < poission.prob(t_s,t_c.tr(x,y,z)) )  then
                tr(x,y,z) = 1;
                t_e.tr(x,y,z) = exprand_t_e(y,T,Vgate);
            end
            x++,y++,z++;
        end
        //Calculate leakage currents resulting from all traps at a given time step
        i=0;
        ΔI_leak.GIDL(t)=0;
        ΔI_leak.gate(t)=0;
        while ( i < N_tr ) do
            if ( x_0 < x ) then //only for traps in overlap between a gate and a drain
                ΔI_leak.GIDL(t) += eq_GIDL(tr_i);
                ΔI_leak.gate(t) += eq_gate_leak(tr_i);
            end
            i++;
        end
        t +=t_s;
    end
    n++;
end
```

Figure 8 A pseudo code of device modeling to generate VRT signals for all cell transistors.

3.1.2 Circuit-Level Simulation Methodology with VRT

To deploy time-varying RTN signals in the leakage current generated from the VRT device model for circuit-level simulations, we propose a gate model. Since a time-varying resistor, as depicted in Figure 9(a), is not applicable to circuit simulation, we exploit a voltage-controlled resistor shown in Figure 9(b). We simply define a new voltage source (e.g., $V_{gate,offset}$) and change the voltage level using a piecewise linear signal that allows the resistor to have a time-varying resistance based on the relationship between the voltage and the resistance defined in the G element. Using a resistor (R_G), RTN in gate leakage is modeled with the time-varying resistance with a control voltage of $V_{gate,offset}$. Similarly, RTN in GIDL is modeled using R_{GIDL}, of which resistance is determined by a user-defined voltage of $V_{GIDL,offset}$. Using a Perl script, we modify a netlist so that every cell transistor has a user-defined voltage source and voltage-controlled resistors. Using the script, we also generate input stimuli for circuit simulations.

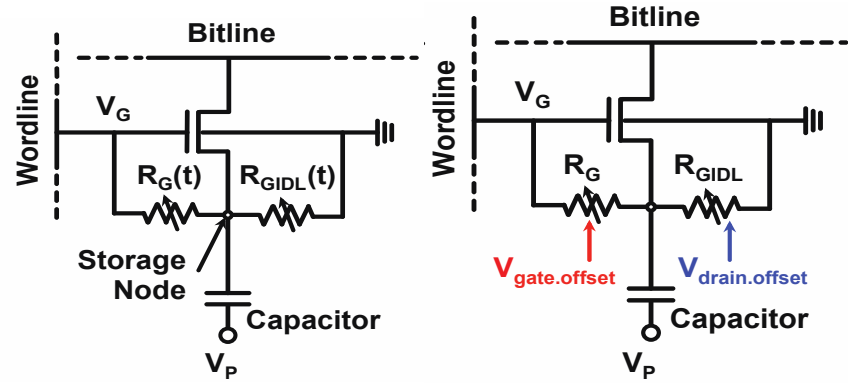
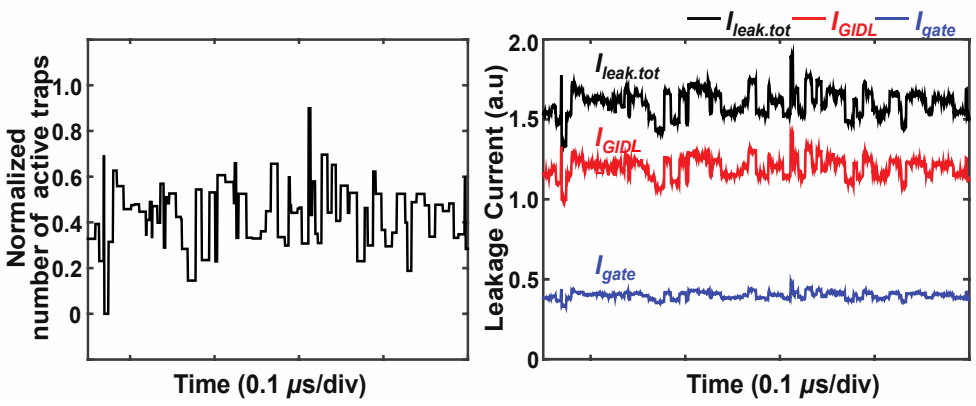


Figure 9 Gate-level models for circuit simulations with (a) time-variant resistors and (b) voltage-controlled resistors.

3.1.3 Simulation Results

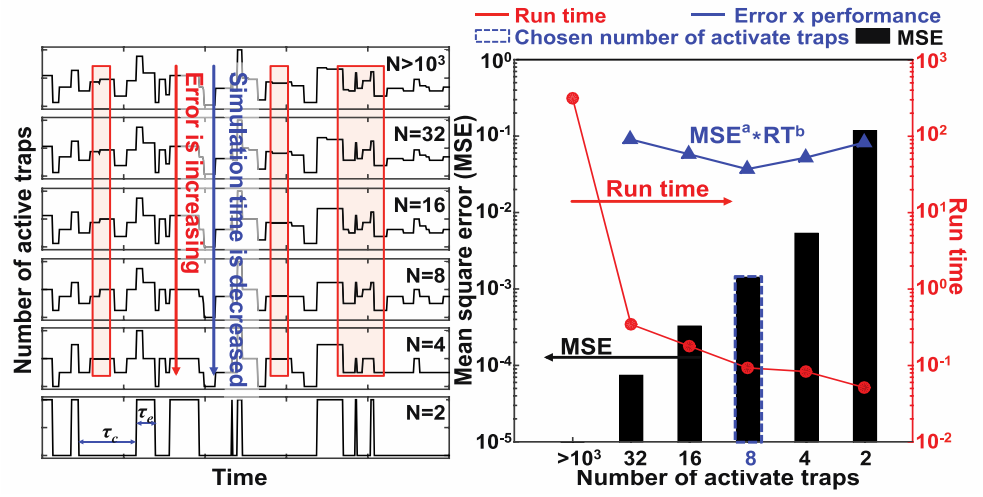
Figure 10 depicts changes in the number of active traps and leakage current fluctuations corresponding to the number of filled traps over time. Despite continuous leakage current levels, we quantize the possible leakage current levels of a transistor for

comparing results with those from experiments in prior work. To optimize the simulator in terms of accuracy and performance, we conduct sensitivity simulations with various numbers of possible leakage states as shown in Figure 11. With fewer leakage levels (*N*), we have more quantization errors and a faster execution time. Based on the error-runtime product with the N>10^3 case as a baseline, we found N=8 to be optimum and use it for the rest of the simulations.



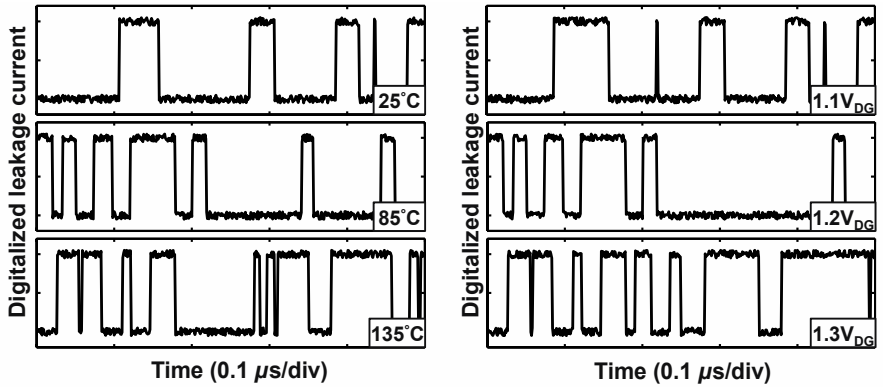
(a) (b)

Figure 10 AVERT simulation results: (a) simulated number of traps filled and (b) time-varying leakage current signals calculated from the number of traps filled.



(a) (b)

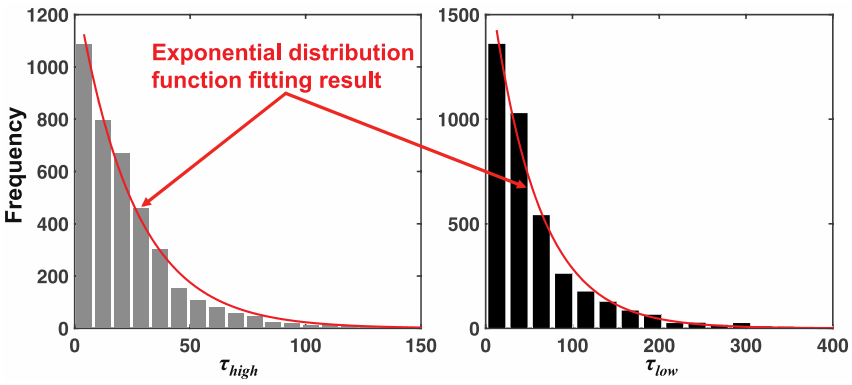
Figure 11 Impact of the number of leakage states (*N*) on (a) the accuracy of simulations and (b) quantization errors and total simulation time with various values of *N*.



(a) (b)

Figure 12 Simulated dependence of (a) temperature with 1.2 V V_{DG} and (b) bias (V_{DG}) on leakage current at 85 ˚C.

To validate our model, we compare our simulation results with prior empirical results, such as measurements of the bias/temperature dependence, results from the Poisson distribution for high and low retention states, and data on the power spectral density. As shown in Figure 12, the bias/temperature dependence on the leakage current generated from AVERT shows more frequent transitions with higher bias and higher temperature, which corresponds well to empirical observations in [1]. The distribution of retention states (τ_{high} and τ_{low}) follows a Poisson distribution, shown in Figure 13, and the power spectral density follows a Lorentzian-type spectrum, depicted in Figure 14. Such results correspond to results from prior experiments [3]. Moreover, such properties match those of the channel RTN of MOSFETs and the GIDL RTN of non-stressed devices [3]. Therefore, AVERT successfully models RTN in leakage currents in DRAMs.



(a) (b)

Figure 13 Distribution of (a) τ_{high} and (b) τ_{low} using AVERT.

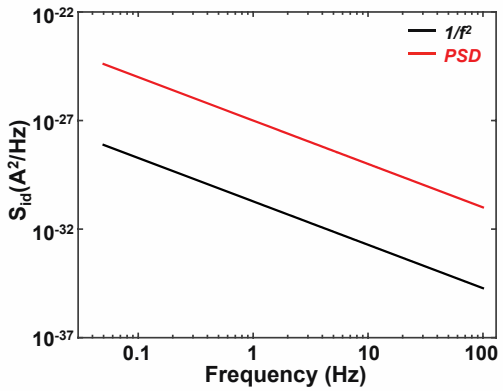


Figure 14 Power spectral density of fluctuation.

We use SPICE with IBM 90nm process design kit (PDK) and the BSIM model to obtain the circuit simulation result for DRAM retention time shown in Figure 15. The upper part of the graph depicts the waveform of a piecewise linear signal to control the resistance of the voltage-controlled resistor of gate leakage and GIDL in a cell transistor. The graph at the bottom shows that charge leaks over time after the storage node is written as '1.' With VRT, the voltage of a storage node randomly fluctuates as leakage randomly varies, and charge leaks faster than it would without VRT.

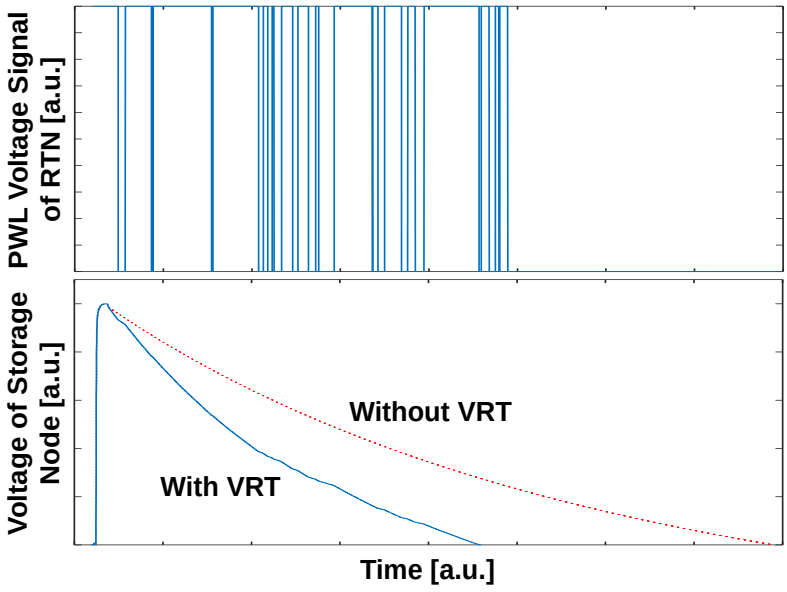


Figure 15 Waveforms of a control-voltage signal of a resistor (upper) and retention time of a cell (bottom).

Figure 16 shows the results of a case study of retention time with VRT. Using AVERT, we obtain the degree of variability in leakage current of DRAM arrays under VRT. Assuming one part per million (ppm) retention errors, a lognormal distribution of retention time with a mean of 10 seconds, and no variability in cell capacitance whose nominal value is 20 fF, we add the variability of VRT to the original retention distribution to obtain the cumulative probability of retention failures with VRT. As a result, the number of retention errors increases from one ppm to 20 ppm because of VRT.

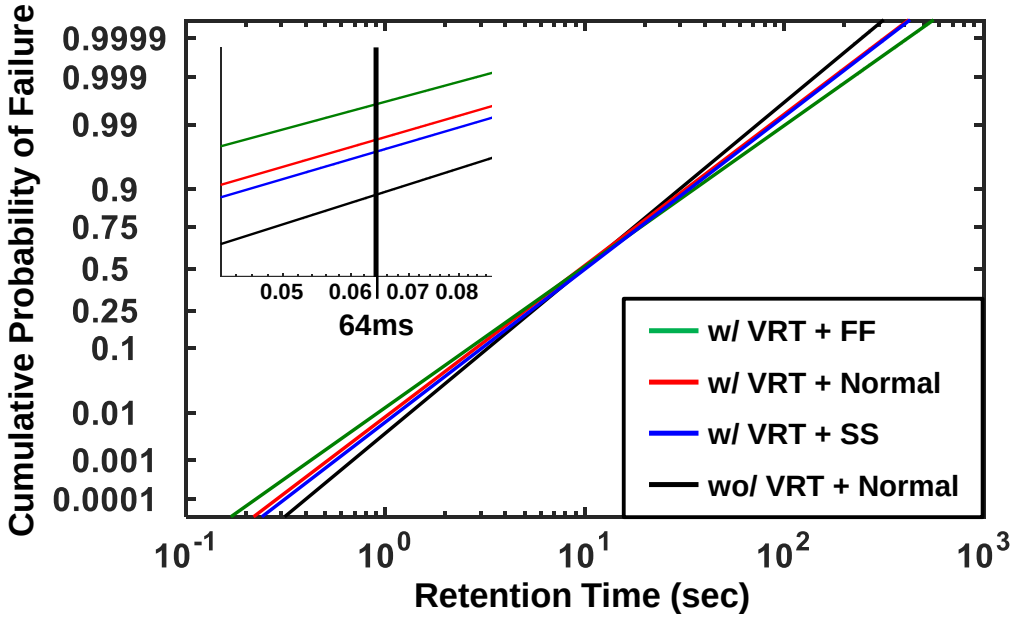


Figure 16 Simulated cumulative probability of errors over retention time based on circuit simulations with and without VRT using AVERT.

3.1.4 An Application Example of AVERT: A Strategy for Testing VRT

Based on the analysis from AVERT, we propose to optimize the total test time of DRAM in the presence of VRT through (1) the optimization of test repetitions and (2) the reduction of the number of bits to be tested. Figure 17 shows the test coverage with various test repetitions, in the presence of VRT, using the same assumptions as in Figure 16. As the number of tests increases, which also increases the test time with a given time step for

testing, the number of retention errors resulting from VRT in DRAM arrays decreases and saturates, as shown in Figure 17(b). Such a trend is useful for optimizing the test time per cell and determining the test coverage. With the optimal test time per cell, we can obtain the degree of variability in retention times of DRAM arrays in the presence of VRT using AVERT, which indicates how frequently a VRT bit can transition from one state to another. By exploiting the degree of the variation, we can reduce the total number of cells that must be re-tested because every VRT cell does not cause a retention error. Only transitions from a higher retention time to a lower retention time below the standard refresh rate (64 ms) such as type (d), depicted in Figure 18(a), result in retention errors. Therefore, we can confine the re-testing of retention time in the presence of VRT to only portions of the DRAM, illustrated in Figure 18(b). The decreased number of bits for testing VRT eventually reduces the overall test time. Assuming that maximum variation of retention times caused by VRT is two seconds, the test interval is 64 ms, and the other assumptions in Figure 16 are the same, the cumulative error rate becomes 6.78 %. Therefore, we require only 6.78 % of the total conventional test time with our proposed testing methods and the reduced number of cells.

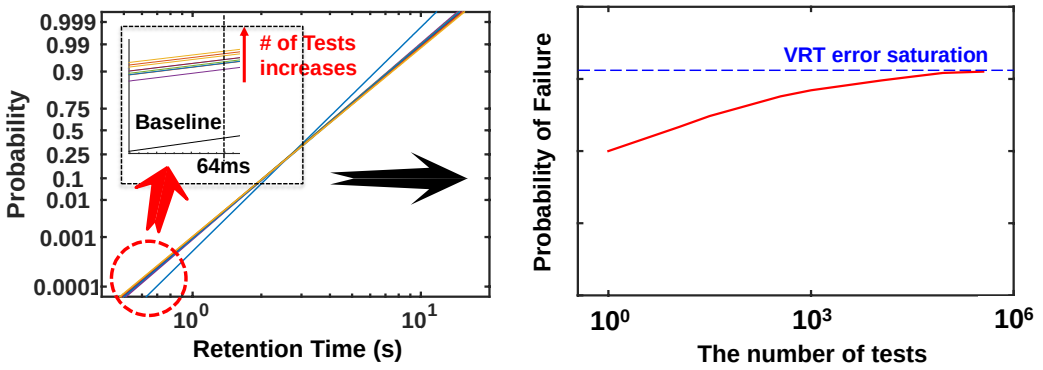


Figure 17 (a) Cumulative probability of retention errors in DRAM arrays with various test repetitions of VRT and (b) probability of retention errors due to VRT with various test repetitions.

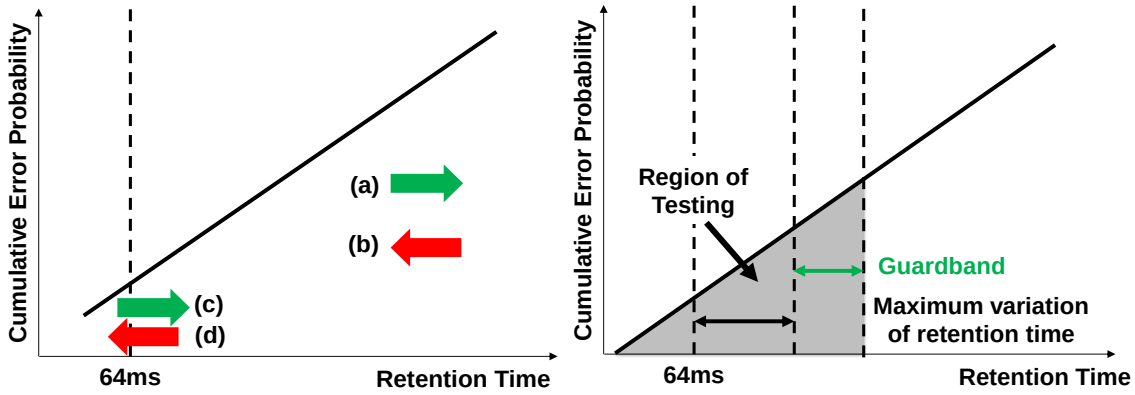


Figure 18 (a) Different types of VRT cells: only type (d) cause a retention error due to the VRT phenomenon and (b) definition of portion of DRAM cells that are of interest based on the maximum retention time transition after model calibration.

3.1.5 VRT Summary

Since random fluctuations in trap-assisted GIDL and trap-assisted gate leakage cause VRT of a DRAM cell, an accurate profile of retention time requires prohibitive efforts for DRAM manufacturers to screen VRT bits during testing. To predict the impact of VRT on devices and support circuit simulations that are aware of VRT, we have proposed AVERT, an elaborate device model of RTN in leakage current and a circuit simulation methodology for variable retention time in DRAMs. AVERT adopts the TD model for better temporal and spatial configurations of traps and calculates leakage current variations of TA-GIDL and TA-GL considering the influence of multiple traps. Our results have shown that AVERT can generate stochastic RTN leakage current signals, fed to circuit simulations with ease after modification of a netlist so that every transistor has a voltage-controlled resistor, the gate-level model proposed with VRT, and the conversion of RTN current leakage into a piecewise linear voltage signal as a user-defined voltage source for resistors. Simulations based on our RTN model have shown reasonable results that closely correspond to results of prior empirical studies. In addition, results from circuit simulation have demonstrated that AVERT enables circuit simulations of DRAM arrays under VRT

for characterizing retention times. AVERT is useful for optimizing the test time per cell and predicting the variability in retention times. Based on such a degree of the variability, AVERT can contribute to reducing the overall or average test time of DRAM cells under VRT.

Prior publications [67] investigated the trapping /detrapping phenomenon of charge in high-k dielectric, and the TD model appears to be valid for high-k dielectrics. Moreover, because prior experimental studies [68] have shown that GIDL of high-k dielectric MOSFETs also exhibits RTN, the method for RTN current calculations based on the trap configuration is also viable with various energy levels for high-k dielectrics. Therefore, although our model was developed based on SiO_2, AVERT is also applicable to the case of high-k materials.

3.2 Architectural Impact of Row Hammering on a Memory System

3.2.1 Typical Workloads

Current workloads can have an activation patterns that target a few DRAM rows frequently. We study the possibility of row hammering using USIMM [69] and use workloads from the memory scheduling championship [70]. We evaluate a system with eight 8 Gb chips [71]. Table 1 shows the configuration for our system. Figure 19 shows the maximum number of activations (activation peak) of a row at the refresh rate of 64 ms for a few PARSEC, SPEC, BIOBENCH and COMMERCIAL benchmarks. Figure 19 shows that workloads have an activation peak of several thousand activations between refreshes. For example, MT-Fluid has 400 *K* activations for a single row within 64 ms. As technology scales these activation peaks can easily surpass the row-hammering threshold.

Table 1 System Configuration (Default of USIMM)

Number of cores Two: 4-wide, 3.2GHz	Number of cores Two: 4-wide, 3.2GHz
Processor ROB size 160	Processor ROB size 160
Cache line size 64 Byte	Cache line size 64 Byte
Last Level Cache 512 KB per core	Last Level Cache 512 KB per core
Memory bus speed 800 MHz	Memory bus speed 800 MHz
Data-rate type three (DDR3) Memory channels 2, each 8 GB DIMM	data-rate type three (DDR3) Memory channels 2, each 8 GB DIMM

3.2.2 Malicious Programs

Malicious programs that frequently activate a given row can easily cause data loss due to row hammering. Unfortunately, such malicious programs are quite easy to write (they are similar to the attack kernels in [72], which can be written in about 10 lines of C code). Thus, for future DRAMs, row hammering not only poses a reliability issue, it also presents a security issue whereby a malicious program can intentionally cause data loss for a co-running program. In this dissertation, we present two hardware solutions to mitigate row hammering in DRAM memories.

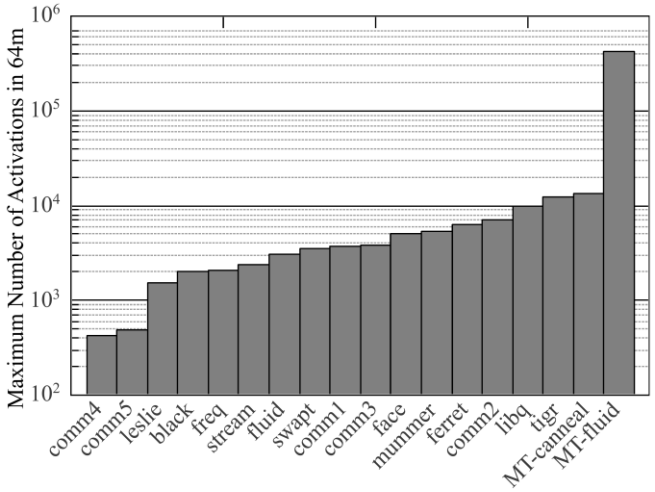


Figure 19 Maximum number of activations for a given row within a time period of one refresh interval (64 ms).

3.3 Circuit and System Simulation of Wearout Errors in Memory

3.3.1 Circuit Simulation of the Impact of Frontend Wearout on eDRAM

We utilize the IBM 90nm PDK in our simulations and exploit the structures and operations of eDRAM based on prior publications from IBM [74],[75]. As depicted in Figure 20(a), eDRAM has a basic cell structure with one transistor and one capacitor. Since BTI and HCI cause shifts in the threshold voltage (V_{th}) of a transistor, we can model the effects of BTI and HCI in eDRAM by changing the V_{th} of a cell transistor in the netlist for SPICE simulations. As for GTDDB, we place a 15 Ω resistor (R_{gate}) between a gate and a wordline to model the gate poly resistance and a 10^3 Ω~10^7 Ω resistor between a gate and either a bitline (R_{gbl}) or a storage node (R_{gsn}) to model paths resulting from GTDDB, illustrated in Figure 20(b). Figure 20(c) shows the architecture of an eDRAM sub-array including a three-transistor micro sense amplifier.

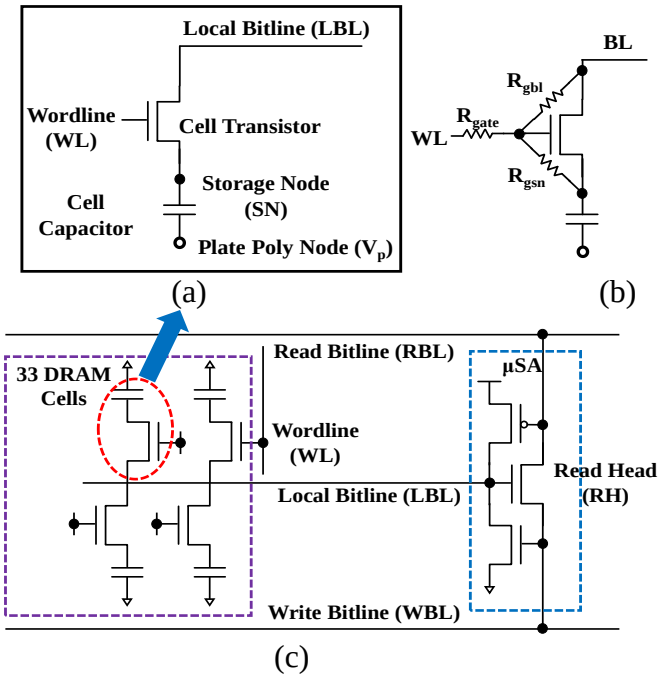


Figure 20 (a) Basic cell structure of eDRAM (1T-1C), (b) a GTDDB model in a cell transistor of eDRAM, and (c) a three-transistor (3T) micro sense amplifier [74].

3.3.1.1 Impact of BTI, HCI, and GTDDB on a Cell Transistor

Figure 21 depicts the effect of the gate voltage on the threshold voltage variation caused by BTI. Since we exploit the charge trapping and detrapping model to understand the impact of BTI [76],[77], such variation relates to the trap energy, duty cycle, and stress time as follows:

$$\mu\left(\Delta V_{th}(t)\right) \propto \varphi\left(T, E_{F,eff}(\alpha)\right) \cdot \left(A + Blog(t)\right), \tag{22}$$

$$\sigma\left(\Delta V_{th}(t)\right) \propto \varphi\left(T, E_{F,eff}(\alpha)\right) \cdot \sqrt{A + Blog(t)}, \tag{23}$$

where A and B are fitting constants, α is the duty cycle, and $\varphi(T,E_F)$ denotes the trap energy distribution as a function of temperature and the Fermi level [77],[78].

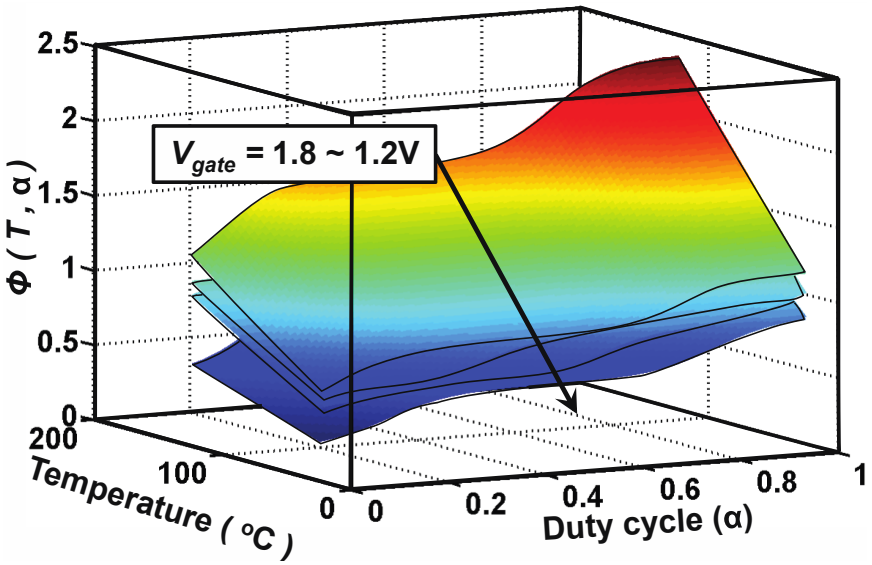


Figure 21 The dependence of the threshold voltage shift on duty cycle, temperature, and gate voltage [75].

Because a cell transistor in this research is an NMOS transistor, as shown in Figure 20, we assume that positive bias temperature instability affects the cell transistor. Since the positive bias on a gate of an NMOS cell transistor is the stress condition under PBTI, a cell

transistor gets stressed from BTI during an active mode of eDRAM operations. Based on prior work [4], we assume that the ratio of stress time, time spent in active mode, to the total operating time is 20 %, which means that the duty cycle is 0.2.

To obtain a sense of how much V_{th} varies in a microprocessor with our assumptions, we exploit simulations from prior work [8] with various workloads, whose experimental setup and simulation methodology are also based on the charge trapping and detrapping model for BTI. Figure 22 shows variations of V_{th} resulting from BTI in a last-level cache (LLC) as a function of stress time. Using the V_{th} distribution, we conduct a Monte Carlo SPICE simulation of eDRAM circuits. Figure 23 illustrates basic write and read operations in an eDRAM cell. During write operations, RBL and WBL turn on the appropriate transistors in the read head, so that LBL is driven to either '1' or '0.' When WL is turned on, the state on the LBL is transferred to the storage node. During read operations, RBL and WBL disconnect LBL from the read head. The state of the storage node is transferred to LBL. If the result is '1,' after LBL is charged up partially, the read head turns on and helps to pull up LBL to '1.'

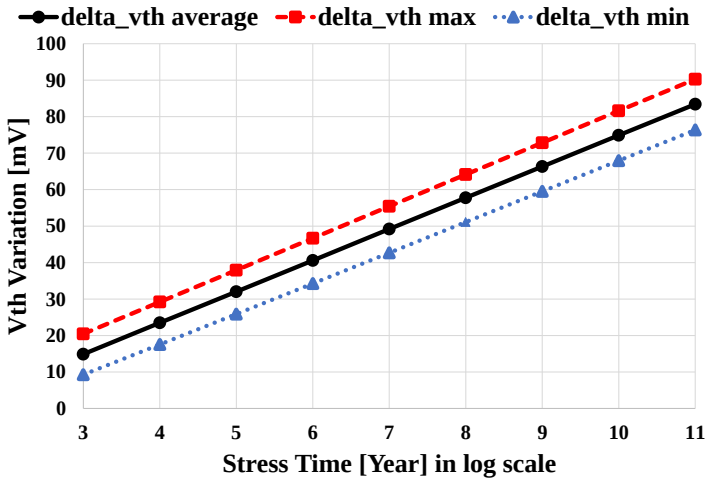


Figure 22 Simulated V_{th} variation caused by BTI, which is generated based on the simulation methodology described in [8].

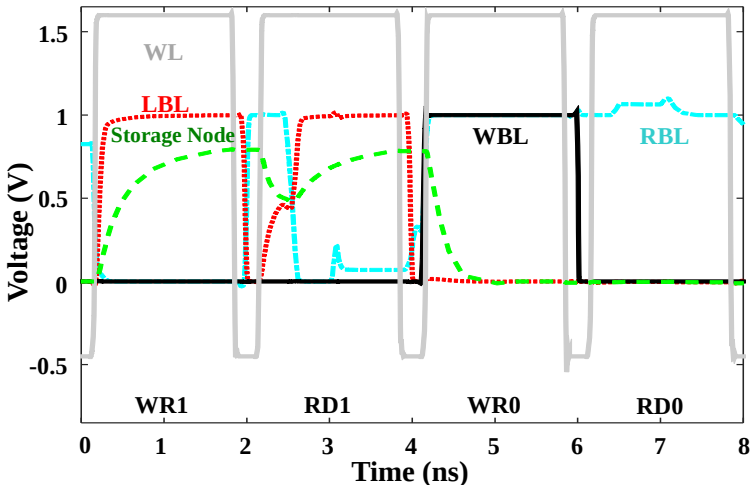


Figure 23 Simulated waveforms of eDRAM basic operations (WR1- RD1-WR0-RD0). WR = write. RD = read.

Figure 24 shows the impact of BTI on a cell transistor with Monte Carlo simulations. The results demonstrate that since the increased V_{th} of a cell transistor degrades its on-current by 9.1 %[1], BTI leads to variation in the voltage of a storage node of eDRAM cells during write '1' operations with a mean shift of 48.1 mV (6.1 %) and a standard variation of 15.4 mV. The lower written voltage level of the storage node also lowers the local bitline voltage level during the read '1' operation. However, no more degradation is found after bitline sense amplification. After all, an NMOS read-head transistor as a single-ended sense amplifier has enough sensing margin to tolerate such degradation caused by BTI because of the high voltage swing on the local bitline owing to the high transfer ratio of 84 %[2].

---

[1] If the voltage between a gate and a source is 1.7 V and V_{th} is 450 mV, the shift in V_{th} from 0 V to approximately 58 mV (at 8 years stress) makes only a 9.1 % decrease in drive current based on the saturation current equation of a MOSFET:

$$\Delta I_{d,sat} / I_{d,sat0} = 1 - \left[(V_{gs} - V_{th}) / (V_{gs} - V_{th0}) \right]^2 .$$

[2] The transfer ratio is

$$C_S / (C_S + C_{LBL})$$

where C_S (18 fF) and C_{LBL} (3.5 fF) are the capacitance of a storage node and a local bitline, respectively [74].

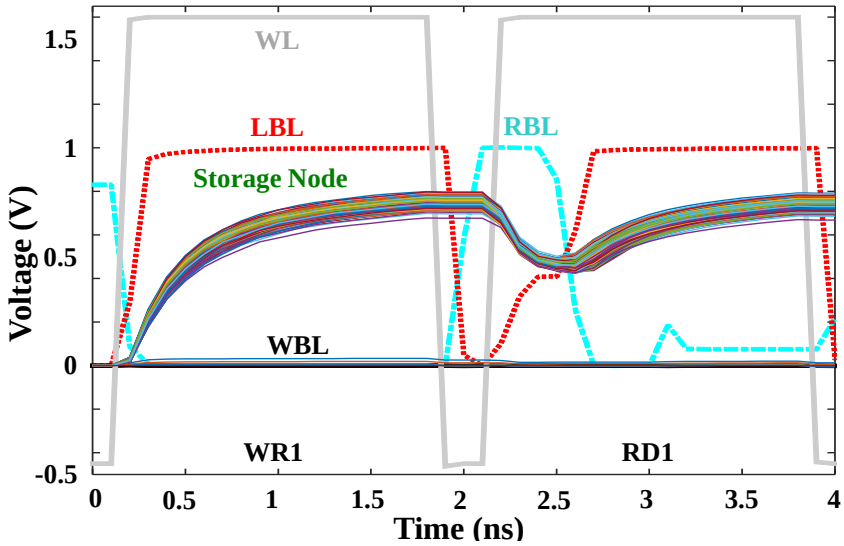


Figure 24 Monte Carlo simulations of the storage node waveform with the increased V_{th} due to BTI.

The V_{th} degradation of a transistor resulting from HCI depends on the stress time (t) as follows [79]:

$$\Delta V_{th} = A \cdot t^{n}, \tag{24}$$

$$A \propto e^{(-\alpha/V_{DS})}, \tag{25}$$

where A and α are fitting constants and V_{DS} is the drain voltage. From the equations, we notice that the shift of the V_{th} resulting from HCI becomes more severe as the drain voltage of a transistor increases. Figure 25 shows the drain voltage of a cell transistor during basic write and read operations. For eDRAM, a write operation of the same data as written in the cell and the write/read operations of data '0' have a zero drain voltage, resulting in no HCI concerns. Hence, only write operations with a data transition from low to high are of interest. However, even during a write operation with a low-to-high data transition, less than 0.6 V is applied to the drain voltage for 250 ps. Since this is too small to generate hot carriers, HCI in a cell transistor is negligible.

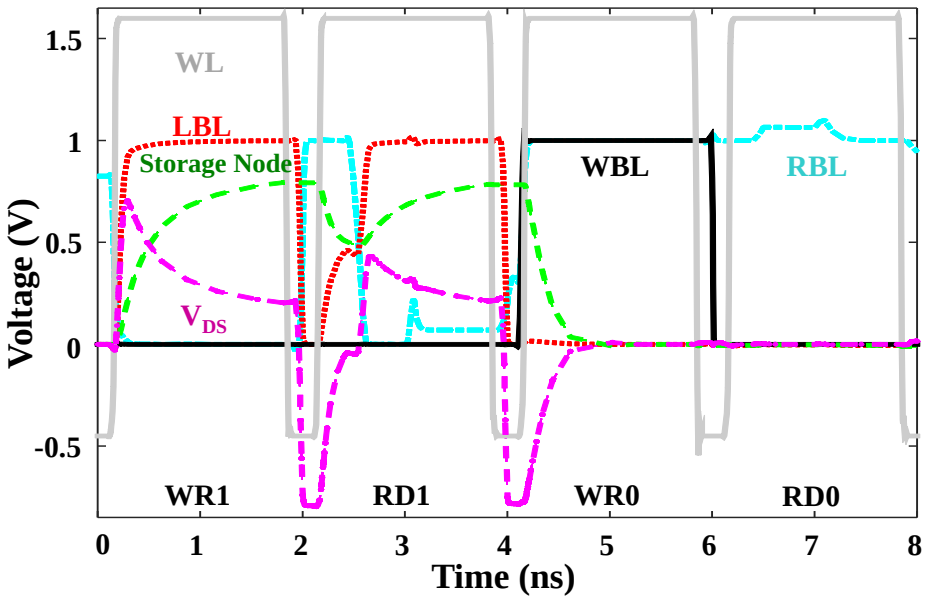


Figure 25 Simulated waveform of the drain voltage (V_{DS}) during eDRAM basic operations.

To understand the effect of GTDDB on a cell transistor of eDRAM, we model paths formed by GTDDB in the gate oxide of a transistor as a resistor whose value decreases as soft-breakdown events occur. For our simulations, we exploit the percolation model illustrated in Figure 26 [80]. Using the PM model, the gate is partitioned into a grid and we statistically generate defects in the dielectric based on the stress conditions, especially the gate voltage. A vertical sequence of defects, a conduction path, forms a leakage path from a gate to either a source or a drain. Once a conduction path is formed, which is referred to as soft breakdown, the resistance between a gate and either a source or a drain reduces. As the number of soft-breakdown events increases as a function of time under stress, hard breakdown occurs. We define hard breakdown when a device no longer functions as a device or when the circuit violates performance specifications [81]. We exploit the number of conduction paths to calculate the resistance of both soft-breakdown and hard-breakdown using the quantum point contact model [82].

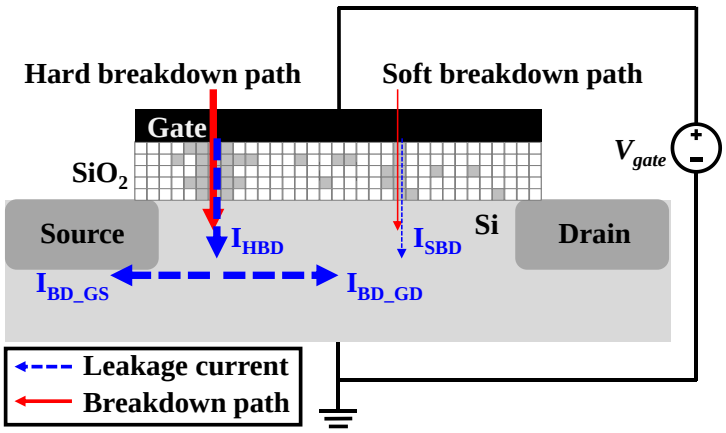


Figure 26 Generation of defects resulting in soft and hard breakdown paths in the dielectric layer (SiO_2) based on the percolation model.

The resistance through the gate oxide is estimated by counting the number of conduction paths. The number of condition paths is a function of the number of defects generated in the percolation model, which is determined by the stress, temperature, and time. We plot the probability of various values of GTDDB resistance as a function of time in Figure 27. In the case that the size of an eDRAM LLC is 32 MB, a stress time of two and half years can cause at least one eDRAM cell within the 32 MB LLC to have 1 MΩ (10^6 Ω) resistance between a gate and either a source or a drain resulting from GTDDB based on simulation results using the percolation model.

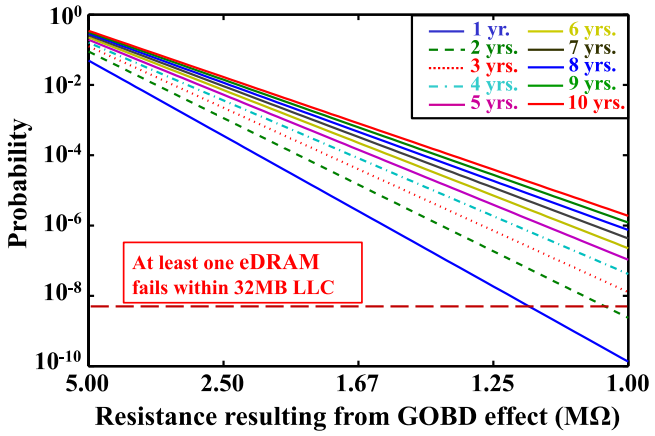


Figure 27 Simulated probability of various breakdown resistances caused by GTDDB as stress time increases. We assume that for a 32 MB LLC at least one eDRAM cell has a 1 MΩ resistance formed by GTDDB after the stress time of two and half years.

To obtain a sense of how small of GTDDB resistance can cause an error in eDRAM basic operations, we simulate a basic sequential column address strobe (CAS) operation of write '1,' read '1,' write '0,' and read '0' with GTDDB resistances from $10^3\,\Omega$ to $10^7\,\Omega$, as shown in Figure 28. Under GTDDB between a gate and a bitline ($GTDDB_{G\text{-}to\text{-}BL}$), a resistance as small as $10^5\,\Omega$ causes read '0' to fail, resulting in a stuck-at-'1' fault, as shown in Figure 28(a). As can be seen from Figure 28(a), the cell state of read '0' changes to '1.' The activated wordline raises the bitline to '1' causing the cell to store a '1,' which is then read as '1' because current flows from the gate to the bitline through the resistor. It can also be seen from Figure 28(a) that the voltage on the storage node relates to the operating frequency. If the period is short, then the read operation functions properly. The fault happens at lower operating frequencies and lower resistances.

Similar to the resistances for $GTDDB_{G\text{-}to\text{-}BL}$, resistances resulting from GTDDB between a gate and a storage node ($GTDDB_{G\text{-}to\text{-}SN}$) as small as $10^5\,\Omega$ also cause a read '0' failure with an operating period of 2 ns, also leading to a stuck-at-'1' fault, as shown in Figure 28(b).

3.3.1.2 Impact of BTI, HCI, and GTDDB on a Cell Capacitor

Note that the variation of the threshold voltage due to BTI relates to the trap energy, the duty cycle, and the stress time in (22) and (23). Since the trap energy increases as the gate voltage increases, an increase in the gate voltage results in more variation in the threshold voltage. Note that the voltage between a cell storage node and a cell plate node is about 0.8 V with data '1' and zero with data '0,' which is much less than that of a cell transistor, which experiences 0.9 V with data '1' and the boosted wordline voltage of 1.7

V with data ‘0.’ Since degradation by BTI depends on bias, the impact of BTI on a cell capacitor is much less severe than on a cell transistor. Hence, we neglect the impact of BTI on a cell capacitor.

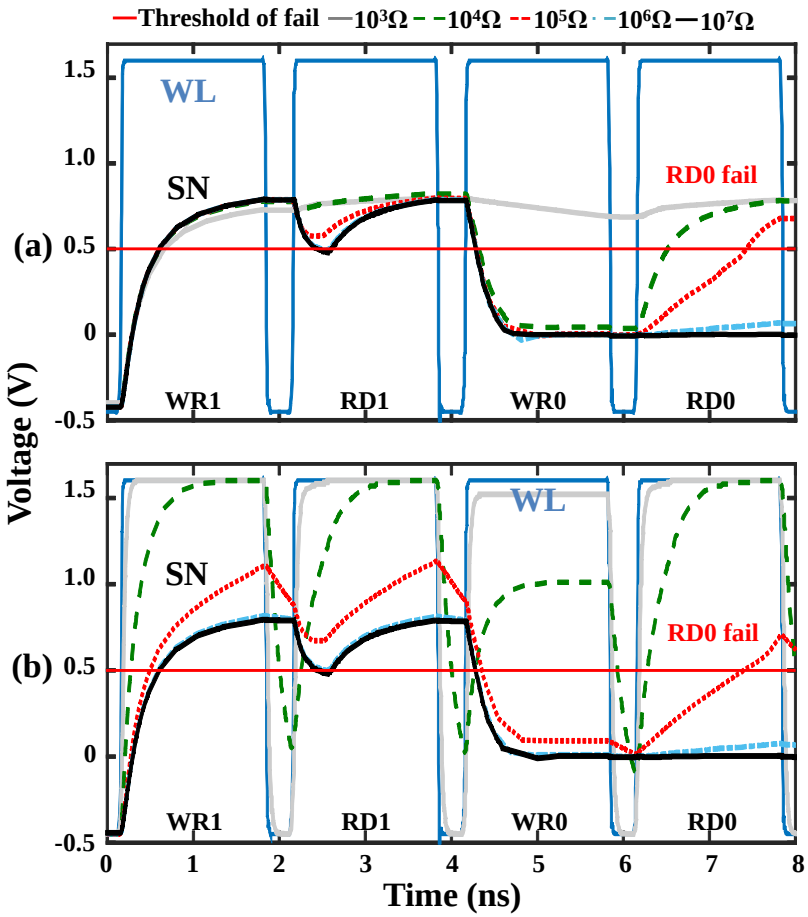


Figure 28 Simulation results of GTDDB (WR1-RD1-WR0-RD0): (a) GTDDB$_{G\text{-}to\text{-}BL}$ and (b) GTDDB$_{G\text{-}to\text{-}SN}$. SN = storage node.

HCI occurs when a hot carrier is formed by the bias between a source and a drain, causing it to penetrate the oxide and fill traps. However, because no drain voltage is applied, the cell capacitor does not suffer from hot carriers. Therefore, we ignore HCI in the cell capacitor.

To estimate lifetime degradation resulting from GTDDB, we exploit equations from [66] and [83]:

$$t_f \propto \left(\frac{1}{WL}\right)^{\frac{1}{\beta}}, \tag{26}$$

$$t_f \propto e^{-\gamma V_{gs}}, \tag{27}$$

where t_f is the time-to-failure at the 63 % cumulative percentile of failure, W and L are the width and the length of a device, respectively, β is the Weibull shape parameter (1.64), γ is the voltage acceleration factor (e.g., 5.6 at 125 °C), and V_{gs} is the gate voltage.

For the calculation of lifetime degradation using (26) and (27), we employ the dimensions of a cell transistor and a cell capacitor for the 65 nm node that are summarized in Table 2 as a case study. The area (WL) of the cell transistor is 0.0144 μm^2 [83] and that of the cell capacitor is 0.205 μm^2 [84]. Because of GTDDB, the lifetime of a cell capacitor is expected to degrade 5.05 times as fast as that of a cell transistor. In addition, since a cell capacitor is stressed during almost all of the execution time although a cell transistor is stressed during write, read, and refresh operations, a cell capacitor degrades four times as fast as a cell transistor, assuming that the portion of the operation time out of total runtime is 20 %. However, note that the gate voltages of 1.7 V for data '0' and 0.9 V for data '1' are applied to a cell transistor, and the bias of 0 V for data '0' and 0.8 V for data '1' are applied to a cell capacitor. Assuming that the signal probability of having data '1' is 35 % in an LLC [4], the reduction in the lifetime of a cell capacitor resulting from GTDDB is 0.0035 times less than that of a cell transistor. All in all, the lifetime degradation resulting from GTDDB in a cell capacitor is only 7.15 % of that in a cell transistor. Since a cell capacitor may have 14 times as long a lifetime as a cell transistor under GTDDB, we neglect the impact of GTDDB on a cell capacitor in this work.

Table 2 Design parameters of eDRAM at the 65nm node

Cell Cap.	Size	Cell Tr.	Cell Tr.
Depth	6.8 μm	Width	120 nm
Opening	98 nm	Length	120 nm
T_{ox}	2.35 nm	T_{ox}	3 nm

3.3.2 A Simulation Methodology for Estimating Memory Reliability Degradation Resulting from Time-Dependent Dielectric Breakdown—TDDB-Emerald

3.3.2.1 Introduction

Because of not only the need for fabrication compatibility with logic fabrication processes, but also the need for fast and reliable performance, SRAM has played an important role as register files and cache memories in modern computer systems. However, continuous scaling down to small dimensions causes new types of wearout challenges in SRAMs. Since MTDDB is one of the growing concerns in terms of reliability, investigating the impact of MTDDB on SRAMs is essential for ensuring reliable operations of SRAMs for next-generation technology nodes.

In this research, we introduce a methodology for Estimating MEmory ReliAbiLity Degradation (TDDB-Emerald) based on device-level time-to-failure models for time-dependent dielectric breakdown mechanisms. Using such models, we estimate the time-to-failure of each potential location in an SRAM cell. With a physical design of the SRAM cell, we define the potential locations of resistive short defects resulting from each time-dependent dielectric breakdown mechanism in an SRAM cell. We implement a bottom-up

approach where we first estimate time-to-failure of each potential defect location from device-level models of the characteristic lifetime resulting from each time-dependent dielectric breakdown mechanism. By combining the time-to-failure of each location, we estimate the lifetime of each SRAM cell in a memory component. Lifetimes of SRAM cells coalesce into lifetimes of each memory functional block, which are further combined to estimate the lifetime of the memory system.

3.3.2.2 Modeling of Time-Dependent Dielectric Breakdown in SRAMs

Figure 29 depicts circuit-level resistance models for potential defects resulting from GTDDB, BTDDB, and MTDDB in an SRAM cell. Time-dependent dielectric breakdown in a transistor causes breakdown in a dielectric, forming a resistive short defect between neighboring conducting materials. Using resistors, we model such possible short defects between nodes, shown in Figure 29.

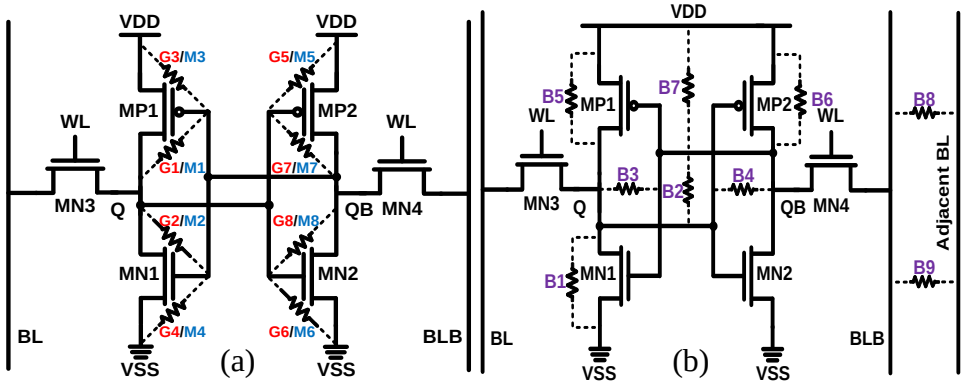


Figure 29 Transistor-level models for potential locations of resistive short defects resulting from time-dependent dielectric breakdown mechanisms: (a) gate dielectric breakdown (G1~G8) and middle-of-the-line dielectric breakdown (M1~M8) and (b) backend-of-the-line dielectric breakdown (B1~B9).

3.3.2.2.1 Time-Dependent Dielectric Breakdown in the Front-end-of-the-Line

We define potential defects resulting from GTDDB in an SRAM cell as resistances G1~G8. Since GTDDB leads to a resistive short defect between the gate and either the source or the drain of a transistor, each transistor can have two resistors, one for the source and the other for the drain. For every node of G1~G8, we check which data condition causes GTDDB stress for a gate dielectric. If data 'High' is stored on node Q in an SRAM cell, the NMOS transistor, MN2, is stressed during the time when logic '1' is stored. Since data 'High' results in data 'Low' at node, QB, the PMOS transistor, MP1, is stressed. For the case with data 'Low' stored on node, QB, in an SRAM cell, the PMOS transistor, MP2, is stressed and degraded by GTDDB. Data 'High' at node Q, driven by data 'Low' at node QB, causes the NMOS transistor, MN1, to be stressed by GTDDB. Once we obtain the information on stress time during operations, we can estimate reliability degradation resulting from GTDDB for each location, G1-G8, in an SRAM cell.

3.3.2.2.2 Time-Dependent Dielectric Breakdown in the Backend-of-the-Line

BTDDB occurs in a dielectric between neighboring metal lines. We investigate possible wearout locations in terms of BTDDB based on an example physical design of an SRAM cell [85]-[86], illustrated in Figure 30. Using resistances B1~B9 shown in Figure 29(b), we model potential short defects caused by BTDDB. For each potential defect site, we examine when a node is stressed resulting from BTDDB. For the case of data 'High' at node, Q, dielectrics associated with B1, B6, and B7 are stressed. However, for the case of data 'Low' at node, Q, only the dielectric associated with B5 is stressed. Regardless of the

data stored at node Q, dielectrics B2, B3, and B4 are stressed. Dielectrics at B8 and B9 are stressed only if the opposite data are applied to the adjacent bitline and bitline bar.

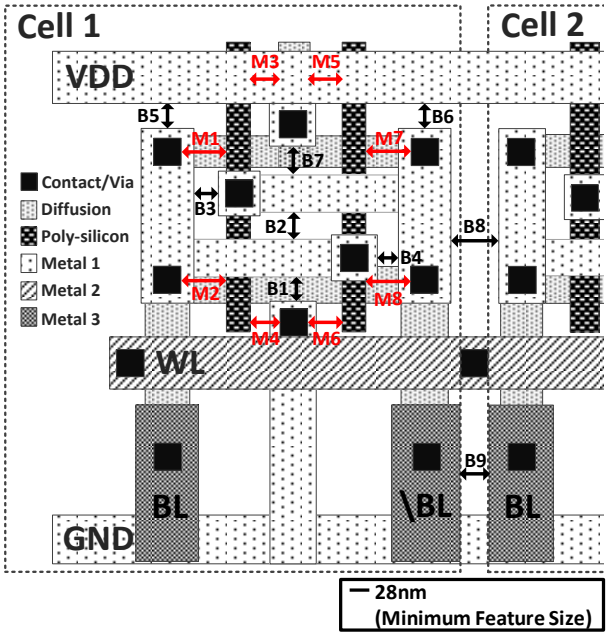


Figure 30 An example of a physical design of an SRAM cell.

3.3.2.2.3 Time-Dependent Dielectric Breakdown in the Middle-of-the-Line

MTDDB occurs in the spacer dielectric between the gate of a transistor and a contact on either the source or the drain of a transistor. After finding contacts that may cause MTDDB in a physical design of an SRAM cell in Figure 30, we determine the possible nodes that are vulnerable to MTDDB. Using resistances M1~M8 shown in Figure 29(a), we model potential short defects caused by MTDDB. For all such defect locations, we examine when a node is stressed due to MTDDB. When data at the node Q is 'High,' dielectric locations, M3 and M6, are stressed. However, when data at the node Q is 'Low,' dielectric locations, M4 and M5, are stressed. Dielectric locations, M1, M2, M7 and M8, are always stressed, independent of the data stored at node Q.

3.3.2.3 Simulation Methodology of Time-Dependent Dielectric Breakdown in SRAMs

Our simulation methodology is built with a bottom-up approach, illustrated in Figure 31. First, using device-level time-to-failure models resulting from each time-dependent dielectric breakdown mechanism in (19)-(29), we estimate the lifetime degradation of each location vulnerable to each wearout mechanism in an SRAM cell. Potential resistive short defect locations for each mechanism in an SRAM cell are based on the observations in the section of 3.3.2.2. In this work, we employ the open-source microprocessor of Leon3 as a case study, whose block diagram is depicted in Figure 32. To estimate the lifetime of a memory functional block (e.g., I-Cache, D-Cache, I-TLB, D-TLB, and Register files), we combine all lifetime distributions from all SRAM cells, which coalesce into the lifetime distributions of each memory block. Finally, we gather all degradation information of each memory unit to estimate the lifetime distribution of the whole memory system.

To calculate the characteristic lifetime of each location using (19)-(29), we exploit fitting constants and Weibull parameters from prior experimental work [1]-[16]. We employ Synopsys 28 nm PDK. To obtain activity profiles, that is, the stress probability and toggle rate, for every SRAM cell, we exploit commercial tools [87] and input the RTL code of a microprocessor [88] and input vectors from open-source benchmarks [89]. After full custom layout based on an example physical design of an SRAM cell [85],[86], we gather geometric information regarding all three time-dependent dielectric breakdown mechanisms.

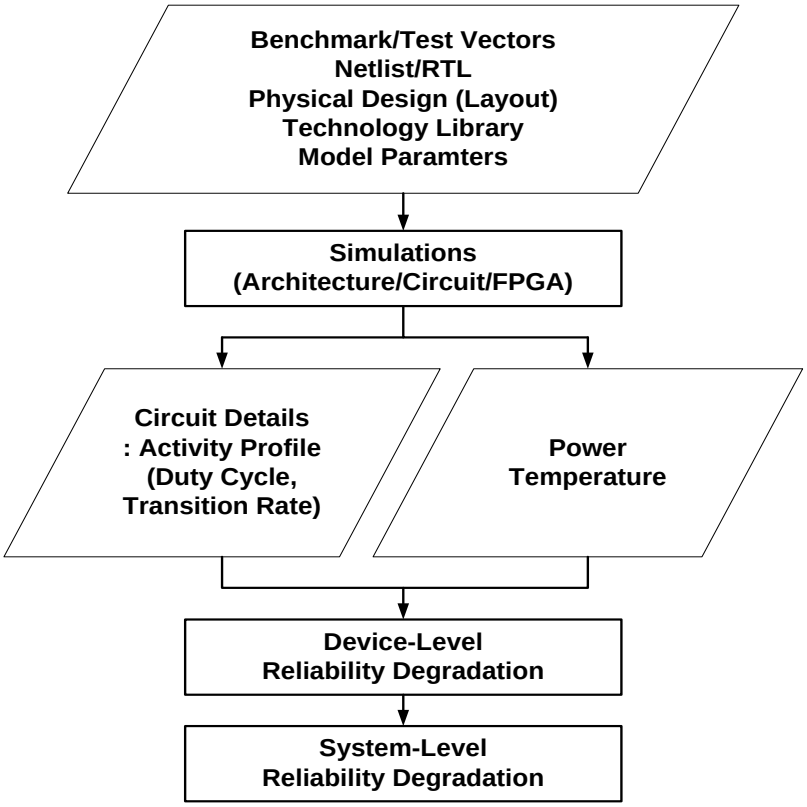


Figure 31 Simulation methodology for estimating memory lifetime distributions. TTF denotes time-to-failure.

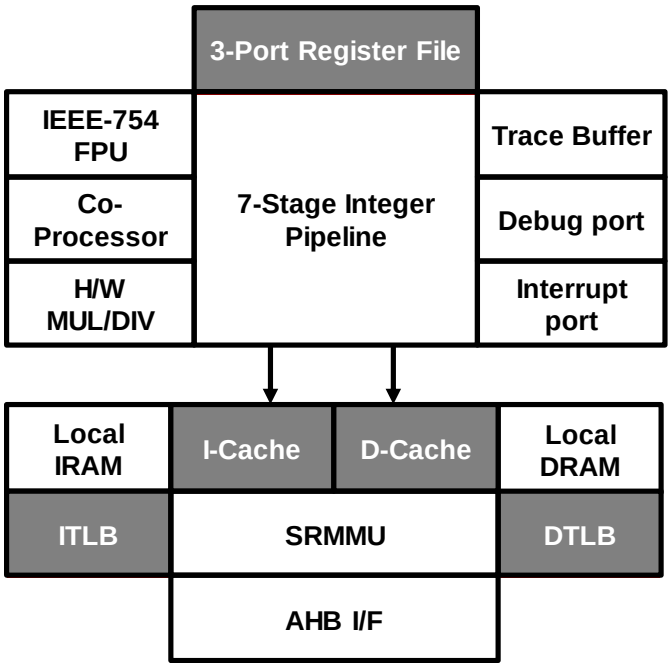


Figure 32 A block diagram of the Leon3 microprocessor [88], which includes approximately 226 *K* SRAM cells. Gray boxes show a memory system in Leon3.

3.3.2.4 Simulation Results

Using the simulation methodology described in the previous section, we simulate the lifetime degradation of a memory system taking all three time-dependent dielectric breakdown mechanisms into account. Among process variations, we investigate the effect of misalignment in the photolithography step on the reliability degradation resulting from MTDDB, depicted in Figure 33. If photolithography misalignment widens the space of the spacer between a contact and the gate of a transistor, which results in a positive misalignment in the graph, such variation enhances the MTDDB reliability of a spacer dielectric, reducing the electrical field between a contact and the gate. However, the opposite direction of misalignment, which presents negative misalignment in the graph, reduces the space between a contact and the gate of a transistor, accelerating the reliability degradation of a spacer dielectric resulting from MTDDB.

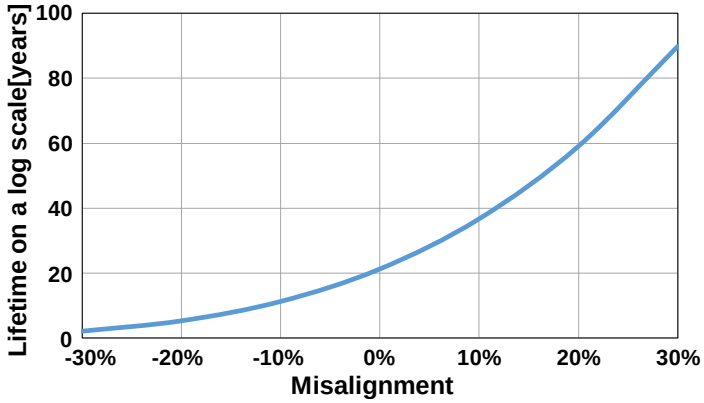


Figure 33 The effect of misalignment in the space between a contact on a source or a drain and a gate of a transistor on the lifetime degradation resulting from MTDDB. A positive sign denotes a wider space caused by misalignment.

The scaling of feature sizes without a reduction in the photolithographic wavelength has resulted in more complex patterning processes, including multiple patterning lithography, such as double patterning for 22 nm six-transistor (6T)-SRAMs [90] and triple patterning for 10 nm and 7 nm technologies [91]. Multiple patterning lithography

aggravates the risk of misalignment error, resulting from offset in opposite directions during different patterning steps. Misalignment not only affects MTDDB, but also BTDDB, when multiple patterning lithography is employed in lithography for the backend-of-the-line.

We break down contributions of each memory functional block to lifetime degradation resulting from time-dependent dielectric breakdown, as shown in Figure 34. Since two cache blocks for instructions and data take up a large percentage of the total area of the microprocessor, they significantly degrade the lifetime of the memory system.

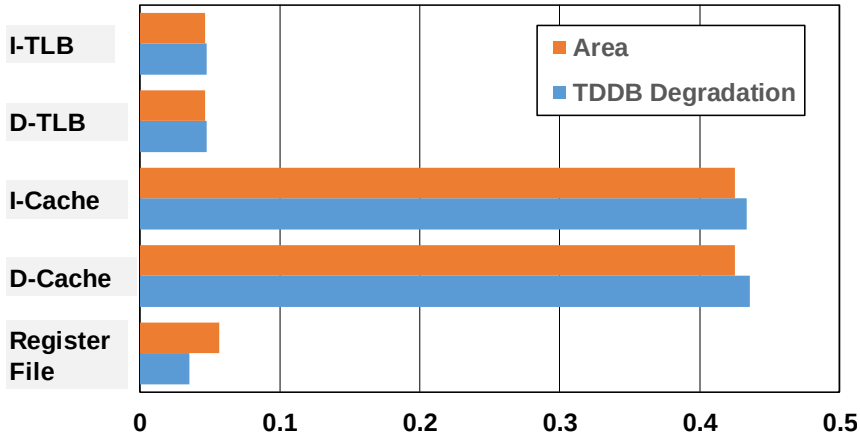


Figure 34 Area ratio of each memory block and contribution breakdown of each memory block to lifetime degradation resulting from time-dependent dielectric breakdown.

In Figure 35, we investigate the impact of each mechanism on the lifetime degradation of each memory functional block. Contributions from each wearout mechanism for all the memory blocks, except the register file, are similar. The difference between the register file and the caches result from the different usage profiles. In Figure 36, we analyze the effects of memory functional blocks on the lifetime degradation resulting from each wearout mechanism. Similar to the results from Figure 34, the main

contributors to wearout caused by all three mechanisms are the two large memory units, the D-Cache and the I-Cache.

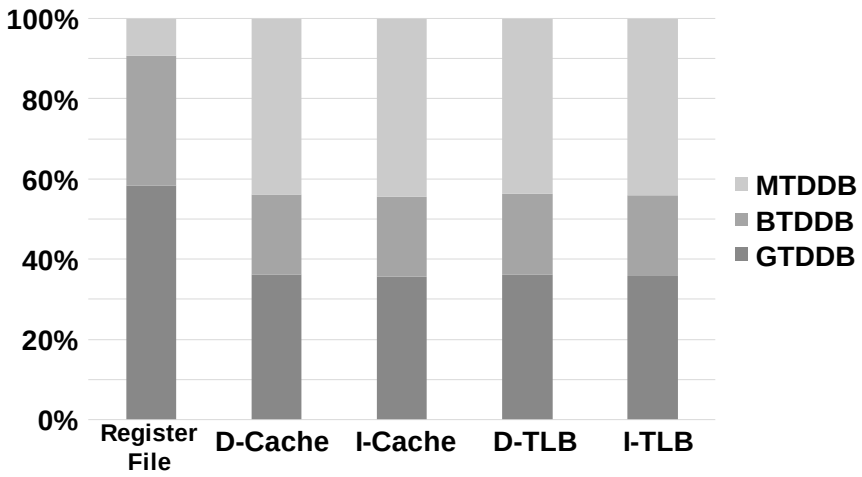


Figure 35 Contribution breakdown of each time-dependent dielectric breakdown mechanism to lifetime degradation in each memory block.

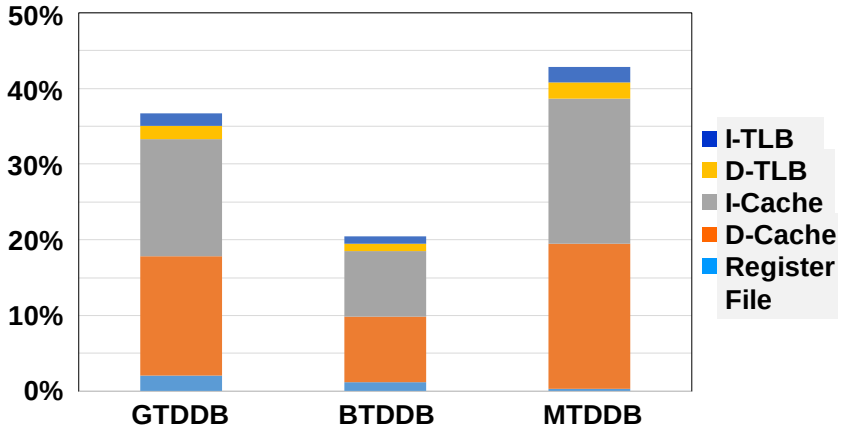


Figure 36 Contribution breakdown of each memory block to lifetime degradation resulting from time-dependent dielectric breakdown due to GTDDB, MTDDB, and BTDDB.

We simulate the cumulative probability of failure resulting from time-dependent dielectric breakdown due to GTDDB, BTDDB, and MTDDB with the general usage scenario [92] at the supply voltage of 1 V and the temperature of 25 $^{\circ}$C, shown in Figure 37. In our case study, since MTDDB dominates the lifetime degradation of the memory system, the overall cumulative probability of failure of the memory system approximates that of MTDDB.

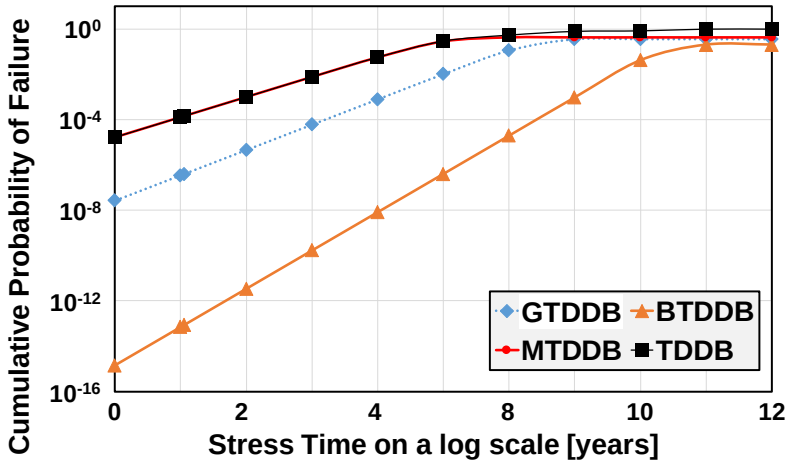


Figure 37 Simulated cumulative probability of failure resulting from each of time-dependent dielectric mechanism, that is, GTDDB, MTDDB, and BTDDB, and the combined effect of all three mechanisms in the memory system of the Leon3 processor.

In (19)-(21), the time-to-failure of each wearout mechanism depends on the voltage and the temperature of the operating conditions. With various stress conditions of various voltages and temperatures, we simulate the time-to-failure of a memory system degraded by the time-dependent dielectric breakdown mechanisms. Figure 38-Figure 41 show the characteristic lifetimes for each wearout mechanism, as this is when 63 % of the population has failed. The characteristic lifetime indicates the sensitivity of the full lifetime distribution to voltage and temperature stress.

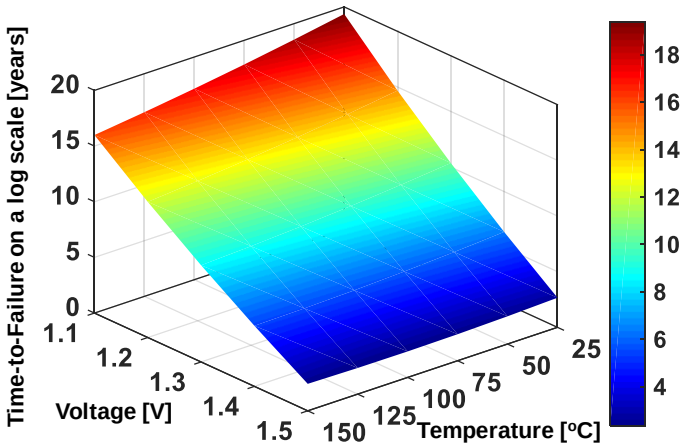


Figure 38 Simulated time-to-failure of the memory system in the Leon3 degraded by GTDDB with various stress conditions for temperature and voltage.

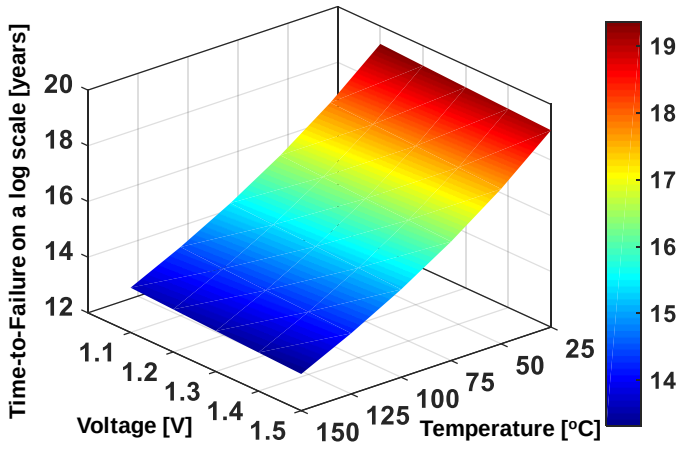


Figure 39 Simulated time-to-failure of the memory system in the Leon3 degraded by BTDDB with various stress conditions for temperature and voltage.

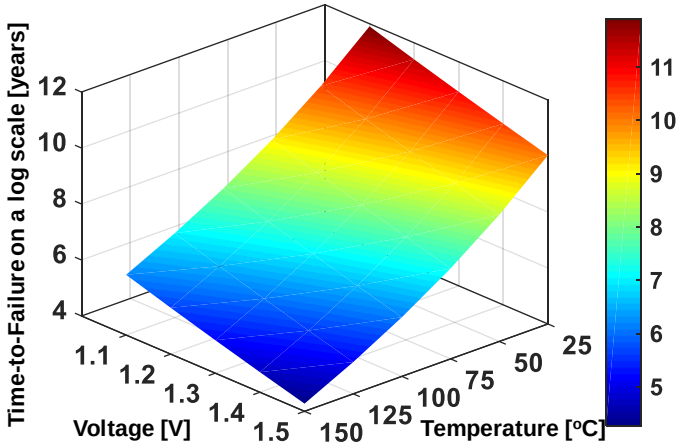


Figure 40 Simulated time-to-failure of the memory system in the Leon3 degraded by MTDDB with various stress conditions for temperature and voltage.

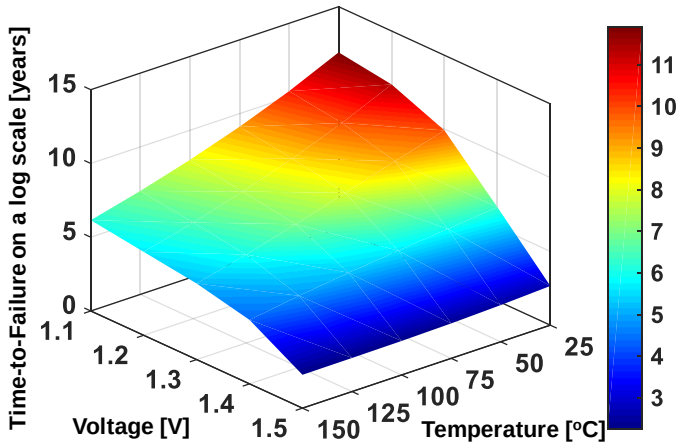


Figure 41 Simulated time-to-failure of the memory system in the Leon3 degraded by time-dependent dielectric breakdown, including GTDDB, BTDDB, and MTDDB, with various stress conditions for temperature and voltage.

At a high gate voltage, GTDDB degrades the lifetime of the memory system severely. However, at a high temperature, we may experience more memory failures resulting from BTDDB. As for MTDDB, both high voltage and high temperature reduce the time-to-failure of the memory system significantly.

By putting all wearout mechanisms together, we obtain the time-to-failure map of the memory system with various stress methodology, after obtaining Weibull parameters of each time-dependent dielectric breakdown mechanism based on the device-level test structures designed for each mechanism, we can estimate the lifetime of the memory system.

It can be seen from the figures that the probability of failure due to GTDDB is dominant at voltages that are higher than 1.4 V. On the other hand, MTDDB is accelerated by both temperature and voltage and is dominant at voltages below 1.4 V and notably at use conditions. BTDDB failures increase with temperature. However, there are no test conditions where BTDDB is dominant.

3.3.2.5 Summary

This work presents a method that estimates a lifetime distribution of a memory system in a microprocessor resulting from time-dependent dielectric breakdown, taking all three time-dependent dielectric breakdown mechanisms into account, i.e. frontend/middle/ backend-of-the-line time-dependent dielectric breakdown. We investigate possible defect locations in an SRAM cell for frontend/middle/backend-of-the-line dielectric breakdown. From the device-level models of the lifetime distribution based on the Weibull distribution for each breakdown mechanism, we estimate the lifetime distribution due to each potential

resistive defect node in an SRAM cell. After combining all lifetime distributions of each node for each wearout mechanism, we estimate the lifetime distribution of each memory functional component, which coalesces into the lifetime distribution of a memory system. Using the open-source microprocessor, Leon3, as a case study, our simulator estimates the lifetime degradation of a memory system resulting from time-dependent dielectric breakdown while taking operating conditions into account.

3.3.3 Investigation Impacts of Wearout on DRAM Cell Behavior

The most common wearout mechanisms are BTI, HCI, GTDDB, MTDDB, BTDDB, EM, and SIV. BTI, HCI, GTDDB, and MTDDB affect the devices, and BTDDB, EM, and SIV affect the interconnect. They all result in permanent faults in field operations. BTI and HCI cause device degradation. GTDDB, MTDDB, and BTDDB eventually result in shorts, while EM and SIV over time cause opens. A picture of the DRAM cell and fault locations is shown in Figure 42-Figure 43.

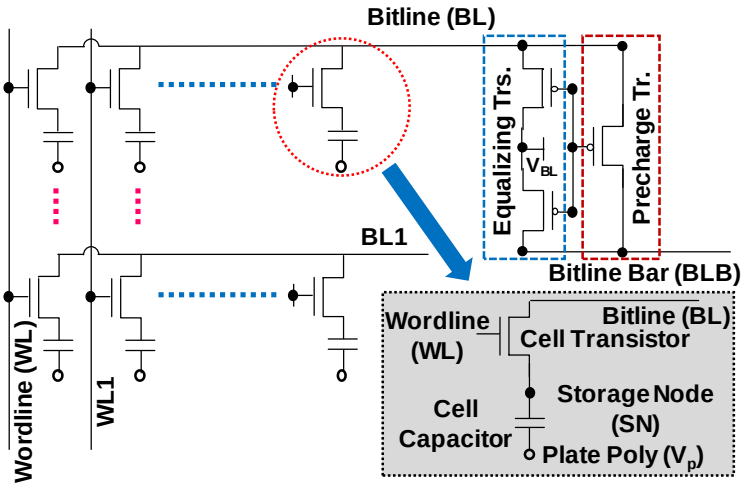


Figure 42 The organization of an DRAM array (The inset illustrates a unit DRAM cell.).

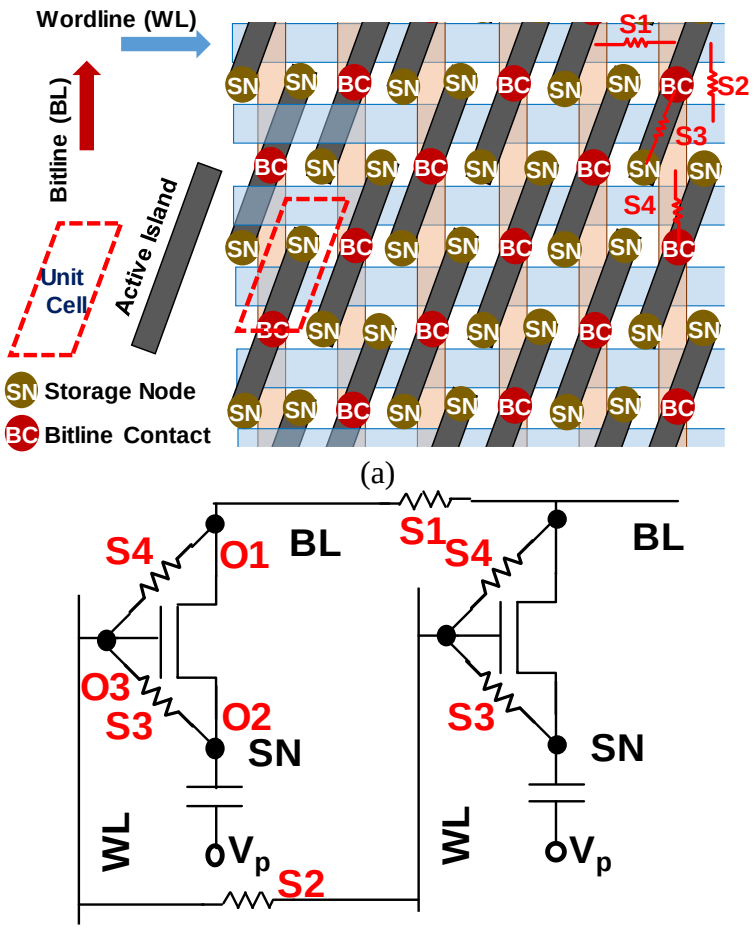


Figure 43 (a) Potential wearout failures in the cell layout of DRAM cells [23] and (b) circuit-level models of wearout failures in DRAM cells. Short failures (S1-S4) and open failures (O1-O3) result from TDDB and SIV/EM, respectively.

Prior studies have investigated the impact of front-end-of-the-line wearout mechanisms such as BTI, HCI, and GTDDB on DRAMs [13]-[18]. BTI and HCI are associated with the gradual degradation of device characteristics, especially the threshold voltage. Such increases in the threshold voltage degrade the operating performances of transistors, such as the cell transistor, the equalizing transistor, and the precharge transistor, degrading the AC characteristics of a DRAM cell, such as retention time (tRET), write recovery time (tWR), and precharge time (tRP) [93]-[95]. Although these failures are dependent on operating conditions, such as temperature, voltage, and operating frequency, over time, depending on the operating environment, they start to behave like stuck-at-faults.

Once degradation is sufficient to violate the AC DRAM specifications, these faults can be detected as hard faults which ECCs repetitively detect in the same memory location.

Since GTDDB in a cell transistor creates a path between a gate to either a source (S3) or a drain (S4) of a cell transistor, such a bridge between nodes of a transistor results in a stuck-at-‘1’ fault in a read ‘0’ operation [95]. Similarly, middle-of-the-line time-dependent dielectric breakdown, which has been investigated as one of the largest contributors to wearout failures in transistors in the 20nm technology node and beyond [12],[44],[96], also can form a bridging fault between a gate and either a source (S3) or a drain (S4) of a cell transistor, leading to a stuck-at-‘1’ fault during a read ‘0’ operation, similar to GTDDB. GTDDB in the cell transistor causes a tRET failure. Over time this fault is detected by ECCs as a stuck-at-zero fault when the gate dielectric loses its integrity resulting from increased leakage from the storage node (S3).

Back-end-of-the-line wearout, such as BTDDB, EM, and SIV, also cause faults in memories [13],[16]-[18],[97]-[98]. Since BTDDB causes a bridging fault between adjacent metal lines, if such a bridge is formed between adjacent wordlines (S2) or bitlines (S1), data from a wordline or a bitline will show multiple errors in a word or in a dataline. ECCs, such SECDED, cannot correct such catastrophic failures in a bank. Other fault-tolerant techniques, such as a chipkill, are necessary for such clustered failures. Note that the target faults of the proposed scheme are not such block failures, but single- or double-bit errors that ECCs can detect.

Since SIV and EM cause open failures (O1-O3) in a via [13],[16]-[18],[97]-[97][98],[99], such wearout causes a stuck-open fault, preventing access to a DRAM cell.

If a via on a bitline (O1) or a wordline (O3) becomes open because of SIV or EM, writing data to a DRAM cell or reading data from a DRAM cell is impossible. Therefore, data read from such a cell is determined by the precharge level of the bitline, mismatch between a bitline and a bitline bar (reference node of a bitline), and an offset of the bitline sense amplifier. Such read data, predetermined by conditions of the read path, exhibits a stuck-at-fault during read operations, since this behavior is repeated for all read operations. Similarly, if there is an open via between the access transistor and the storage cell (O2), the storage capacitor becomes very small and cannot influence the bitline, again causing stuck-at behavior determined by parasitics in the read path.

Many prior studies have also demonstrated the prevalence of soft errors, caused by neutron-induced cosmic rays with high energy, B fission induced by neutrons, and alpha particles, in memories [100]. Therefore, this work takes both soft and hard faults into account in detecting and identifying faults in DRAMs during field operations.

Wearout progresses with time. This means that incrementally, wearout faults are single bit faults. Over time, multiple bit faults will develop. However, this work aims to incrementally find locations of single bit errors and fix them, prior to the development of multiple bit failures.

CHAPTER 4. MITIGATION TECHNIQUES FOR ERRORS IN MEMORY

4.1 Row-Hammering-Aware Memory System

Mitigation of row hammering can be done by sending a proactive activation to the victim rows before the target row crosses the row-hammering threshold. Such a proactive activation acts as a refresh command for the victim rows, and refreshes the contents of these rows, thereby preventing data loss due to the activity of the neighboring rows. We propose two schemes, counter-based row activation (CRA) and probabilistic row activation (PRA). CRA scheme tracks activations for each row and provides guaranteed mitigation, whereas PRA is a probabilistic scheme that avoids storage overhead and yet provides highly robust mitigation.

4.1.1 Counter-Based Row Activation

The CRA scheme maintains a row activation counter (RAC) for every row to keep track of the number of activations to each row. These activation counters are incremented when the row is activated and cleared when mitigation is performed. As soon as the number of activations of a target row is equal to the row-hammering threshold, the victim rows associated with the given row gets activated. Such proactive dummy activations of victim rows refresh their data and prevent data corruption caused by row hammering. To cope with future memory systems with lower row-hammering threshold, this work employs 2-byte (16 bits) long activation counter per row that can count up to 64 K activations per refresh cycle. For an 8 GB memory system with one million rows, the total size of the

counters for all rows will be 2 MB. However, it is impractical to devote multi megabyte of on-chip SRAM storage for storing the counter of CRA. Instead, we propose a CRA implementation that stores the counters in a reserved area in the DRAM (0.0375 percent of main memory reserved for the counters). To mitigate performance penalty of counter accesses we employ a dedicated counter cache on chip. A memory controller checks the counter-cache for activation counters and caches them from the reserved area. The reserved area is only accessed on a counter-cache miss for activation counters. Every access to the reserved area brings a cache line with activation counters for 32 contiguous rows, which ensures high locality in the counter-cache. In steady state, rows with frequent accesses and high locality will have their counters cached. A memory controller increments the counter of row activated and clears the counter after a row is refreshed or on mitigation. Figure 44 shows the sequence of events for CRA.

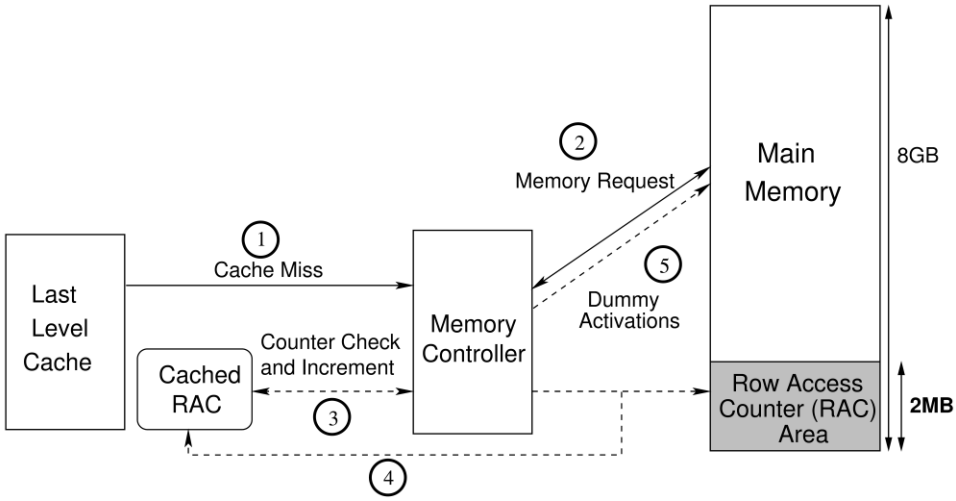


Figure 44 Sequence of memory operations with CRA.

Figure 45 shows the performance impact of CRA on execution time, as the size of the on-chip counter cache is varied. Even though we used a 32 *K* threshold for this study, the performance degradation stems mainly from the memory accesses for the counters. With a counter cache of 128 KB, CRA scheme has less than 0.5 percent performance

degradation, while ensuring that the victim rows get refreshed before the target row reaches the threshold.

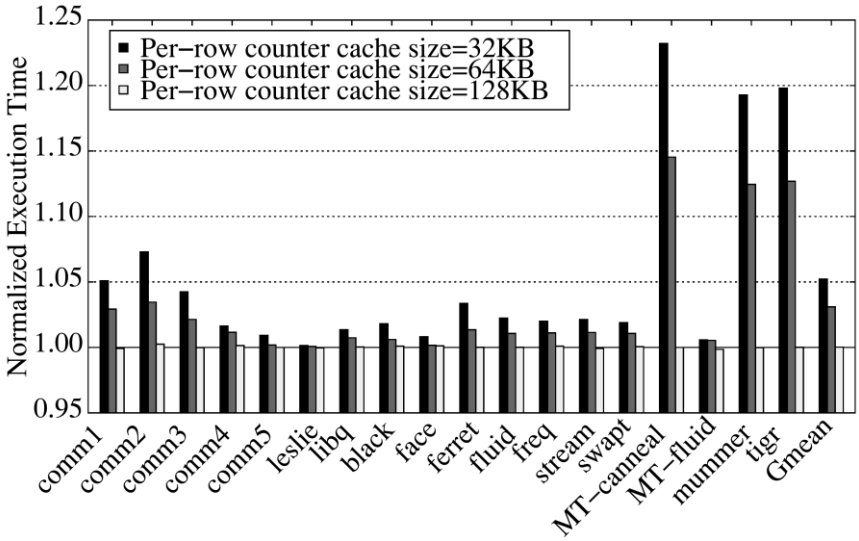


Figure 45 Impact on execution time from CRA (Row Hammering threshold of 32 *K*).

4.1.2 Probabilistic Row Activation

The PRA scheme avoids the storage overhead of the CRA scheme by obviating any tracking structures. Instead, it performs a row activation of the neighboring rows with a small probability every time a given row is accessed. For example, if the probability of activation is set to 0.1 percent, then for each row activation, the memory controller consults a random number generator to find if the proactive activations must be issued. If so, the memory controller proactively inserts activations for the two neighboring rows for the row being accessed. The key insight for PRA is that hammered rows will have frequent activations and hence are highly likely to get selected for probabilistic mitigation.

4.1.2.1 Analysis

Let us consider a system that performs dummy activations of neighboring rows on each access with a probability N. The probability of a target row not being activated after

M activations in total within a refresh rate and resulting in a potential system failure is given by (28):

$$P_{error} = (1 - N)^M = (1 - N)^{\frac{1}{N}MN}. \quad (28)$$

If we perform these proactive dummy activations with a very small probability, then such activations have a negligible effect on performance.

Since $\lim_{N \to 0}(1 - N)^{\frac{1}{N}} = e^{-1}$ for very small values of *N*, we deduce (29) from (28):

$$P_{error} = e^{-M \cdot N}. \quad (29)$$

From (29), the probability that the system will have no errors is given by (30):

$$P_{no-error} = 1 - e^{-M \cdot N}. \quad (30)$$

Let the system during its lifetime have K such instances. The probability of having no failures in the entire lifetime of the system is given by (31):

$$P_{no-failure} = (1 - e^{-M \cdot N})^k. \quad (31)$$

Subsequently, the probability of at least having one failure during its total runtime is given by (32):

$$P_{failure} = 1 - (1 - e^{-M \cdot N})^k. \quad (32)$$

In the worst case, if N = 0.1 %, M = 32*K* (row hammering threshold) and for runtime of 10 years, K ≈ 25 billion; then (33) and (34) shows that the probability of data loss with PRA would be 1 in ten million, over a period of 10 years:

$$P_{failure} = 1 - (1 - e^{-32K \times 0.001})^{25 \times 10^9}. \quad (33)$$

Since $(1-n)^x \approx 1 - nx$ for small n_x, (33) degenerates to (34):

$$P_{failure} = \frac{25 \times 10^9}{e^{32}} \approx 10^{-7}. \quad (34)$$

The failure probability could be made as low as 10^{-120}, with a N=1 %. Thus, even though PRA does not require any storage structures, it can still provide very robust protection against row hammering, even at very small row hammering threshold.

4.1.2.2 Results

Figure 46 shows the impact of PRA on execution time, as the probability of dummy activation is increased from 1 to 0.01 percent. Higher probability of issuing dummy activation (N) degrades the performance of most workloads. For example, probability of activation of 1 percent results in 2 percent increase of activations. On an average, this increases the execution time by around 2 percent. As the probability of issuing dummy activation is reduced to 0.1 percent, the performance degradation is negligible (<0.2 percent on average). Thus, PRA avoids both storage and performance overhead of CRA, and still provides robust mitigation to row hammering.

4.1.3 Row Hammering Summary

Frequent activations to a row can influence neighboring DRAM cells and cause data corruption due to row hammering. To mitigate row hammering, we propose two architectural solutions: counter-based row activation and probabilistic row activation. We

expect row hammering to become even more severe for future memory chips. An experimental study [73] demonstrates that row hammering is indeed prevalent in modern DRAM chips, and their measured threshold of 128 *K* is consistent with our 130 *K* based on our theoretical model. Technology scaling accelerates row hammering and makes DRAM vulnerable to other sources of errors. We show that architectural solutions can help mitigate such errors efficiently and thus help with DRAM scaling.

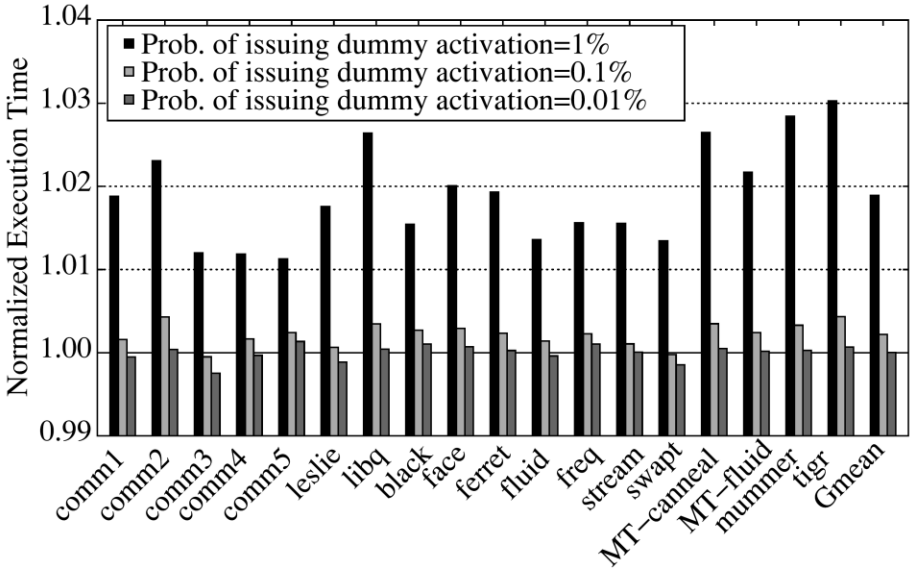


Figure 46 Impact on execution time from PRA.

4.2 Wearout Errors in Memory Systems—ECC-Aspirin

4.2.1 Introduction

DRAM has been widely used as main memory in computer systems because of its high capacity at a low cost. To achieve low cost per bit, DRAMs continue to scale down to smaller feature sizes. As a result, core arrays of a memory have become extremely dense. Because of such high densities, cells are more vulnerable to reliability issues. Reduced scaling of the supply voltage, less than the scaling of the feature sizes, raises further reliability concerns in comparison with previous fabrication process technology

generations. For the enhancement of memory yield, DRAM manufacturers have employed hard repair techniques that replace bad cells in memory arrays with good cells using spare rows and columns. Before DRAM products are released, errors are screened and repaired with spare rows and columns. However, as DRAMs operate in the field, both soft and hard errors also occur, since memory cells are stressed and wear out over time. As a result, errors resulting from wearout, unless corrected or repaired, may cause failure of memories in the field. Such system failures result in degradation of system performances and the cost of replacing faulty DIMMs.

Recent field studies on DRAM errors have reported several remarkable observations [26]-[28]. First, these large-scale field studies have found that contrary to common assumptions that soft errors dominate hard errors, hard errors outnumber soft errors. Second, they have also found that errors are correlated in time and space. Although single-bit errors are dominant, correctable errors are highly likely to be followed by errors with the same address and in the same column and row. Consequently, such errors may advance to uncorrectable errors that cause an expensive system crash requiring the replacement of DIMMs and downtime for the system. Moreover, architectural studies have shown that memory access patterns are not uniformly random. Instead, a few rows are excessively accessed [29],[30]. Such observations proclaim that solely relying on SECDED ECCs cannot sustain a memory system with aging errors in the field. This argument is supported by plotting the reliability of a 2 Gbyte ECC-DIMM with 25,000 to 75,000 FITs per billion hours of operation per Mbit in a DIMM [26], shown in Figure 47. From the simulation results, after 1.85 years of operation with 75,000 FITs per Mbit in a DIMM, 50 % of DIMMs fail. Although ECCs correct a single-bit error in a word, the

presence of a corrected error in a word increases the probability of a memory failure caused by a double-bit error in a word that ECCs cannot cover. Therefore, allowing ECCs to correct single-bit errors is not the ultimate solution for extending the lifetime and the reliability of DRAMs.

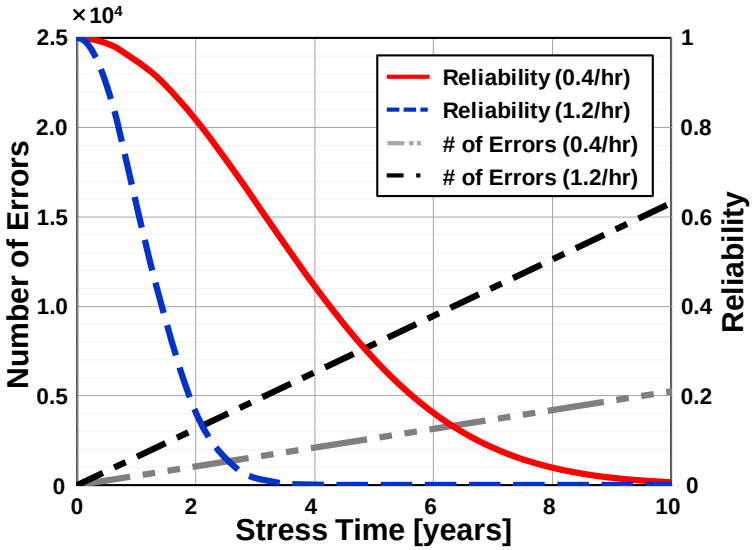


Figure 47 The number of errors and reliability of a 2 Gbyte ECC-DIMM with 25,000 and 75,000 FITs per Mbit (corresponding to 0.4 and 1.2 errors per hour). Reliability is the probability of survival [101].

Studies that have targeted embedded memories [23][24] and main memories [31] have proposed combined schemes with ECCs and repair, i.e. with BIST and/or BISR. Such studies have also investigated a PPR scheme for DRAMs. Such a repair scheme is used for manufacturing errors, not for aging errors that occur during field operations [24]. Moreover, because current BIST and/or BISR technology cannot cover all DRAM test requirements, DRAM manufacturers still rely on automatic test equipment (ATE). As a result, BIST and BISR have not been implemented in commercial DRAMs. This work focuses on standard DRAMs—main memories that do not contain either BIST or BISR.

Assuming that a memory system without either BIST or BISR is an ECC-DIMM with SECDED ECCs, we propose an ECC-assisted post-package repair scheme for aging

errors in DRAMs. We exploit ECCs for detecting faults in field operations. For correctable errors[3], to minimize performance degradation and storage overhead, we propose a detection method that temporarily stores fail addresses of correctable errors occurring during field operations in the LLC. When detecting faults, we propose an ECC-activated fault-type identification of both correctable and uncorrectable errors using the proposed test flow of *Read-Write-Read-Compare* and *Read-Invert-Write-Read-Compare*, which distinguishes hard errors from soft errors and finds the error locations in DRAM chips with a finer granularity than a word. After identifying the error type and location, we propose a method in which a memory controller finds an available storage in the anti-fuse arrays of DRAMs.

By frequently eliminating correctable errors that indicate potential locations of uncorrectable errors in a word, our ECC-assisted post-package repair scheme enhances the reliability of a memory system. Using a simulator that estimates the yield of a memory accounting for the errors in manufacturing and their repair schemes, we demonstrate the feasibility of our proposed post-package repair scheme with negligible overhead in terms of area and performance.

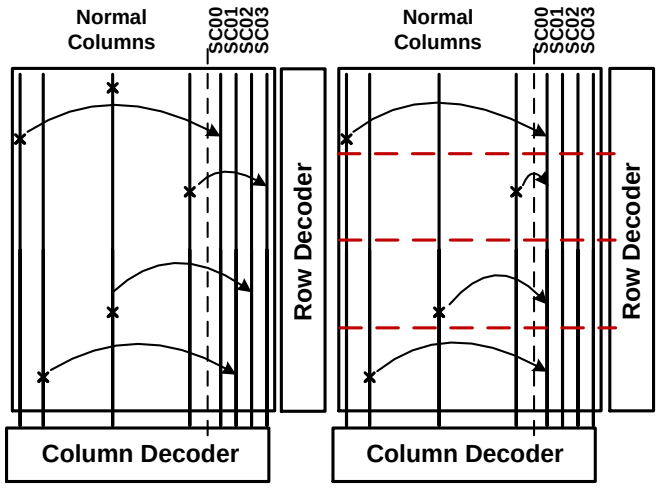
4.2.2 A Conventional DRAM Repair Scheme

4.2.2.1 Spare rows and columns

Figure 48(a) illustrates a conventional wafer-level repair scheme that exploits spare rows and columns to replace faulty rows and columns. The comparator logic that checks whether accessed row/column addresses match faulty row/column addresses stored in fuses

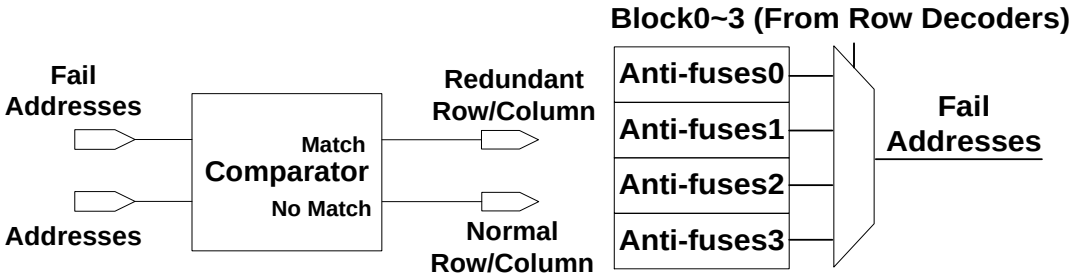
[3]We classify errors in a word into two types: a correctable error (CE), that is, a single-bit error in a word that SECDED ECCs can correct and an uncorrectable error (UE), that is, a double-bit error in a word that SECDED ECCs can detect, but not correct.

is shown in Figure 49(a). If the addresses match, the driving circuit in the comparator activates the corresponding redundant row/column instead of the normal row/column. Otherwise, the driving circuit activates the normal row/column. Since infant failures may still include clustered errors, column or row failures, and peripheral logic errors, spare rows and columns are effective for wafer-level repair.



(a) (b)

Figure 48 (a) A conventional DRAM repair scheme using spare rows and columns and (b) segmented spare columns, which are redundant resources with a finer granularity than that of conventional spare columns (e.g., four segments) [101].



(a) (b)

Figure 49 (a) Comparator logic for conventional remapping and (b) circuit-level modifications of comparator logic for segmented column redundancy for use in post-package repair [101].

4.2.2.2 Segmented column repair with finer granularity

Based on observations that most aging errors are single bits, but such errors are likely to have multiple failures with the same address and in the same column or row [26],

to replace such aging errors, we require redundant resources whose granularity is finer than that of the conventional spare rows and columns for single-bit aging errors. Since spare rows and columns sacrifice several thousand good cells even for repairing only a single-bit error in a row/column, we propose to use redundancies with finer granularity to enable DRAMs to efficiently repair aging errors in a column. To do so, we exploit the segmented spare column technique [102]. In this technique, multiple errors in various arrays share a spare column, as illustrated in Figure 48(b) which shows four segments in each row as an example. Using a spare row, we can replace multiple errors in various segments at the same time. Such segmentation enhances the efficiency of the spare column by at most a factor of four.

Since a row decoder already has segment information that can be obtained after decoding several bits of either the most or least significant row-address bits, depending on the row address distribution of a memory architecture, we can implement a segmented spare column scheme by just feeding several decoded row addresses to column decoders. The row address specifies the portion of the segmented redundancy used to correct faults in the specified column address. The fail addresses of several columns that share a spare column are multiplexed in the column decoder to activate a spare column, as illustrated in Figure 49(b). In the example, the decoded row addresses indicate the segmented block information (Block0~3). If a portion of a column is faulty, the column decoder selects the appropriate redundant segment. We store various fail column addresses in anti-fuse arrays (Anti-fuses0~3) which are selected and delivered to the comparator according to the controlling values of the segment (Block0~3). The rest of the operations are the same as conventional remapping.

4.2.2.3 Storage Elements for Fail Addresses

To store fail addresses for remapping, DRAMs employ laser fuses. For programming fuses, an external laser is the source of energy that cuts the metal fuse. Since metal fuses should be exposed to air for the laser cut, a laser fuse is nonprogrammable after packaging. Thus, a laser fuse cannot be used as a storage element for fail addresses in post-package repair. For DRAMs that exploit laser fuses for wafer-level repair, another type of fail-address storage programmable after packaging is required for use with redundant resources for post-package repair of aging errors. Recently, one-time programmable memories, such as e-fuse, EPROM, and anti-fuse, have been introduced in DRAMs for both wafer-level and post-package repair [33]. To incorporate redundant resources dedicated to manufacturing- level repair into redundant resources for PPR, we assume that anti-fuse arrays [103] are available for both wafer-level repair and PPR in DRAMs.

4.2.3 The Proposed Scheme for Post-Package Repair

The purpose of this work is to develop a post-package repair scheme for a main memory system that conventionally is not equipped with a memory BISR scheme that consists of three basic blocks, BIST, BIRA, and address reconfiguration (AR) for remapping, as illustrated in Figure 50. Similar to a BISR scheme, the proposed scheme has three steps, which are fault detection (FD), fault identification (FI), and fault repair (FR).

We exploit ECCs instead of BIST for in-field fault detection. In the fault-identification step, since faults detected by ECCs contain both stuck-at faults and intermittent faults, we propose a method that distinguishes hard errors from soft ones and finds the locations of faults instead of using BIRA. By introducing sequences of

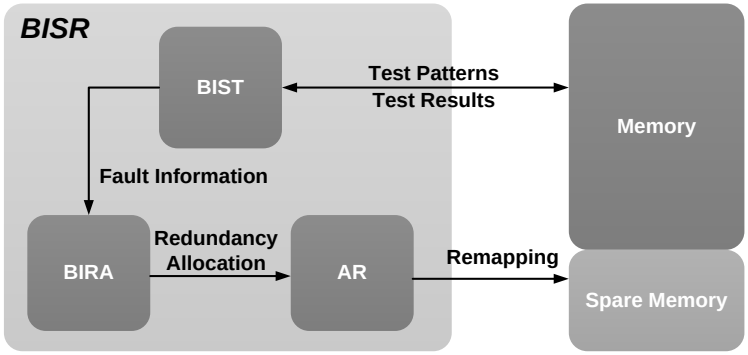


Figure 50 Conventional post-package repair using BISR including BIST, BIRA, and AR for embedded memories.

commands—*Read-Write-Read-Compare* for CEs and *Read-Invert-Write-Read-Compare* for UEs—and a small circuit-level modification in DRAM chips, we identify types and locations of faults in a word with a finer granularity than an eight-byte word. After identifying the type and the location of a fault, the DRAM stores the fail address in the anti-fuse for remapping in post-package repair. To reduce the area overhead of storage for remapping, the remaining manufacturing-level redundancies are used to repair aging errors in PPR. Figure 51 briefly illustrates the proposed PPR scheme.

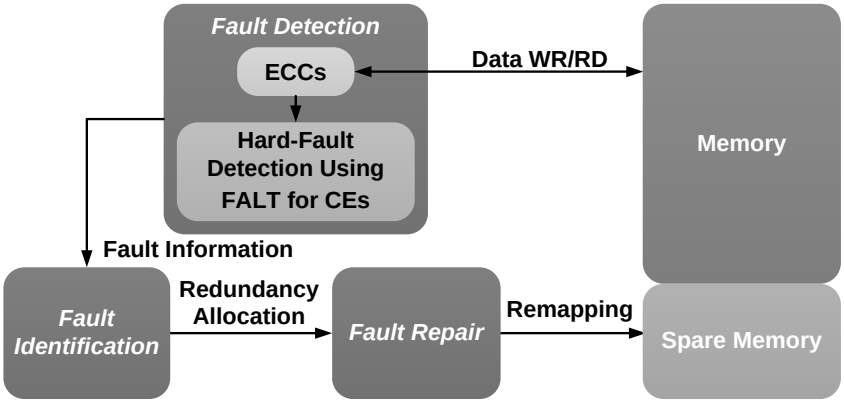


Figure 51 Similar to BISR, the proposed PPR scheme without BISR for DRAMs consists of three steps: fault detection, fault identification, and fault repair.

4.2.3.1 Fault Detection Using ECCs During Field Operations

Figure 52 depicts the conventional and proposed test/repair flow and field operations with ECCs. Once SECDED ECCs correct a single-bit error in a word, the system

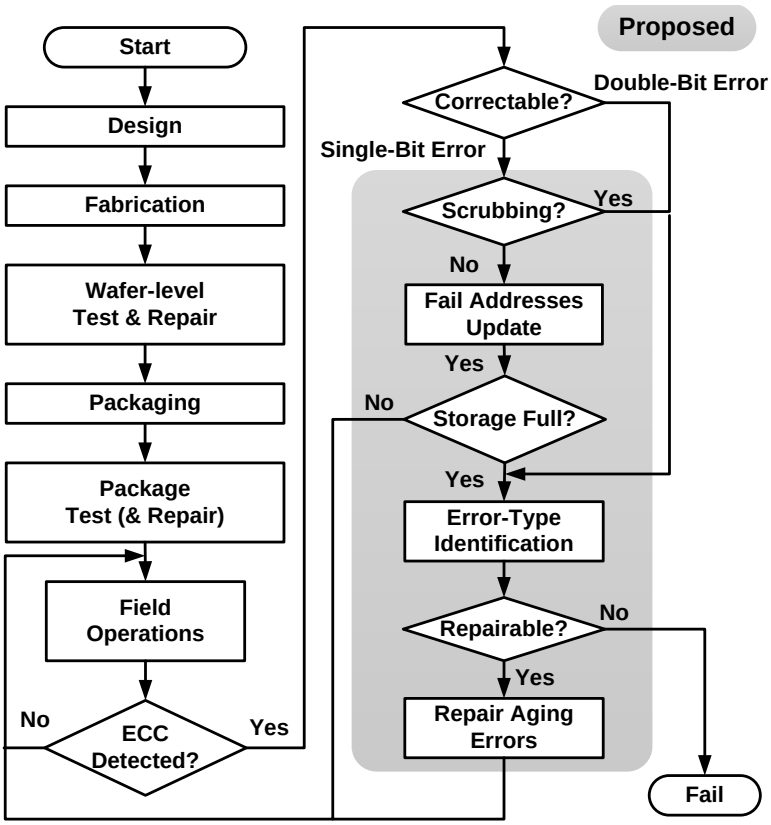


Figure 52 Conventional flow of manufacturing-level test/repair and field operations with ECCs for DRAMs and the proposed test and repair flow for aging errors in DRAMs.

returns to normal operations. Since we want to ensure that system performance does not degrade, interrupting normal operations for identifying whether an error is a hard or soft error is undesirable in a fault detection step. Therefore, to minimize performance overheads for fault-type identification in a fault detection step without interrupting normal operations, we propose a hard-fault detection method (HFD) where a memory controller checks if an error repeats with the same address. Since a soft error rarely occurs twice in the same location, repetition is a good indicator of a hard failure or an aging error.

Figure 53 illustrates the scheme to identify the fail address of a correctable hard error. To increase confidence in determining if an error is a hard failure, a fail-address lookup table (FALT) is allocated in the last-level cache for enabling the memory controller

to store and to track potential fail addresses with a counter. This work presents a case study with a two-bit counter. Whenever a memory controller corrects an error in a word, it looks up the address in the FALT and searches for an entry with the fail address that matches the current fail address. If the memory controller finds a match between the current fail address and one in the lookup table, it increases the counter bit of the entry, whose maximum counter bit is 11_2.

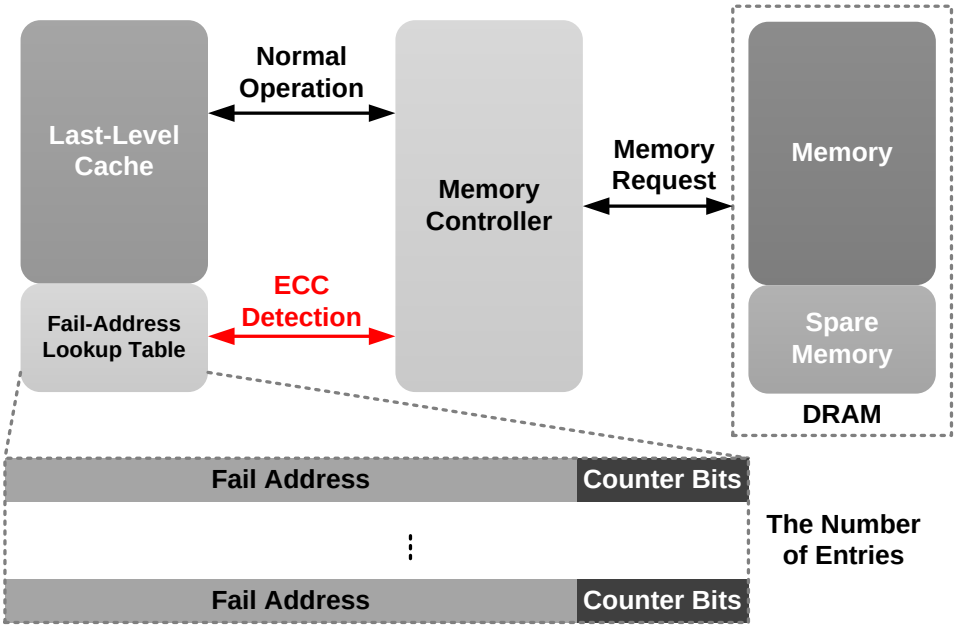


Figure 53 Detecting errors using ECCs and employing the fail-address lookup table (FALT) for temporarily storing the fail addresses of correctable errors occurring during in-field operations until the fault-identification step is activated when the FALT is full.

If there is no match, the memory controller seeks an available entry whose counter bit is 00_2. If the memory controller finds one, it stores the address in the entry and increases its counter bit to 01_2. If the memory controller fails to find either a match or available space whose counter bit is 00_2 in the FALT, it replaces an entry with the smallest counter bit or the least-recently used (LRU) entry if several entries have the same counter bit with the current fail address.

Although we have implemented the LRU strategy as the replacement policy, since we have implemented fail-address lookup table implemented in the last-level cache in this research, any of various replacement policies used in caches, such as random, first-in first-out (FIFO), and least-frequently used (LFU) [104] can be also employed. Because the hit rate of the fail-address lookup table depends on failure characteristics, such as a failure rate and the temporal/spatial locality of failure pattern caused by aging errors, the choice of a replacement policy for the FALT impacts the optimization of the performance of the proposed scheme.

If all entries in the FALT have counter bits larger than one (01_2), we consider that the FALT is full in this research. It is possible to set a higher trigger value such as 10_2, where the FALT is full when each fault has been seen at least twice.

To check if an entry is full or not, we set a trigger value of the counter to 01_2 in our case study, which indicates at least one occurrence of each fault in each word whose address is listed in the table. As a result, we conservatively evaluate the performance overhead of the proposed scheme in chapter 4.2.5, and a trigger value to 01_2 for a fail-address lookup table with 15 entries causes the fail-address lookup table to be full within the 12-hour interval for memory scrubbing, during which approximately 15 correctable errors occur at the worst case failure rate of 75,000 FITs per Mbit, a failure rate based on recent field studies in data centers [26]-[28]. For a fixed failure rate of aging errors, the trigger value and the number of entries in the FALT determines the frequency of repair and the probability of multiple errors in a word, which relates to the performance overhead of the proposed scheme and risk of system failure due to double bit failures within a testing interval.

When the FALT is full, this event activates the next fault-identification step. In other words, the initiation of fault identification is event-driven activation. After sending fail addresses in the FALT to the DRAM, the memory controller evicts all entries of the FALT. Therefore, in event-driven activation, the number of entries for storing fail addresses determines the frequency of post-package repair. For a case in which the number of aging errors increases exponentially as stress time increases, the interval between event-driven repairs becomes shorter as the DRAM wears out. If the failure rate is constant, as with the exponential distribution, intervals between PPR events will be constant.

Unlike correcting a single-bit error in a word, detecting a double error in a word results in a system crash. Before the system powers down or reboots, the memory controller performs a diagnosis of such faults and repairs them. Since all program execution may halt with the detection of double-bit errors that ECCs cannot correct, temporary storage, such as a FALT for correctable errors, is not required for double-bit errors. With a faulty address in which a double-bit error occurs, the memory controller immediately initiates the fault-identification step.

If a memory system exploits memory scrubbing, widely used for mitigating soft errors, once a single-bit error is corrected or a double-bit error is detected during scrubbing, the fault identification step begins. Since memory scrubbing is performed after a pause in normal operations, hard-fault detection using a FALT is not required for both CEs and UEs. Therefore, the memory controller proceeds directly to the fault-identification step. The proposed schemes during both in-field and memory scrubbing operations are depicted in Figure 54 and Figure 55, respectively.

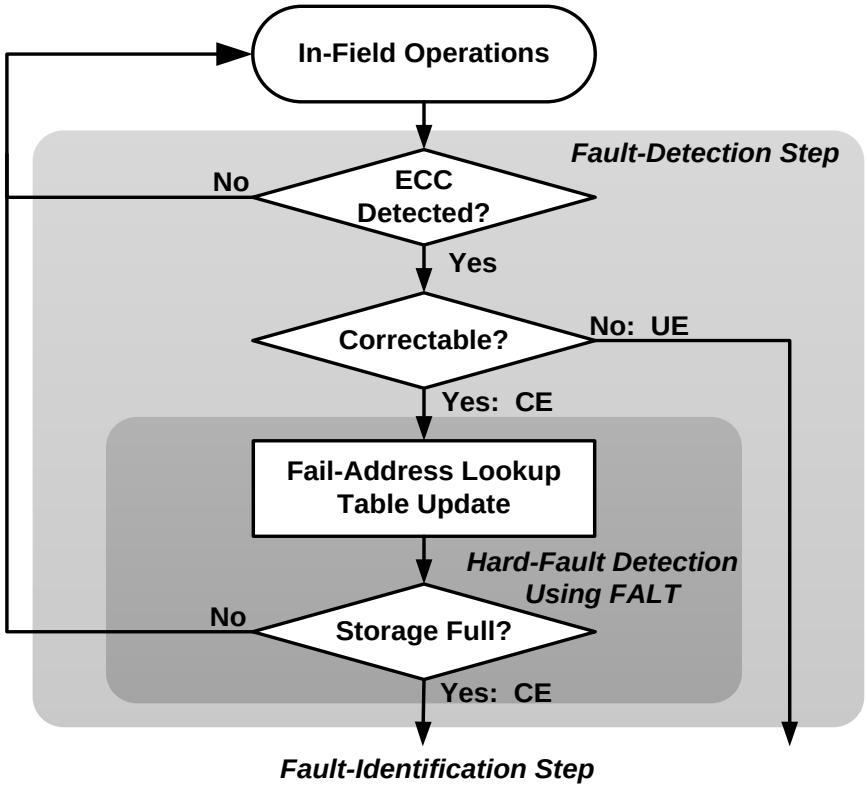


Figure 54 The flow chart of the proposed fault-detection scheme during in-field normal operations.

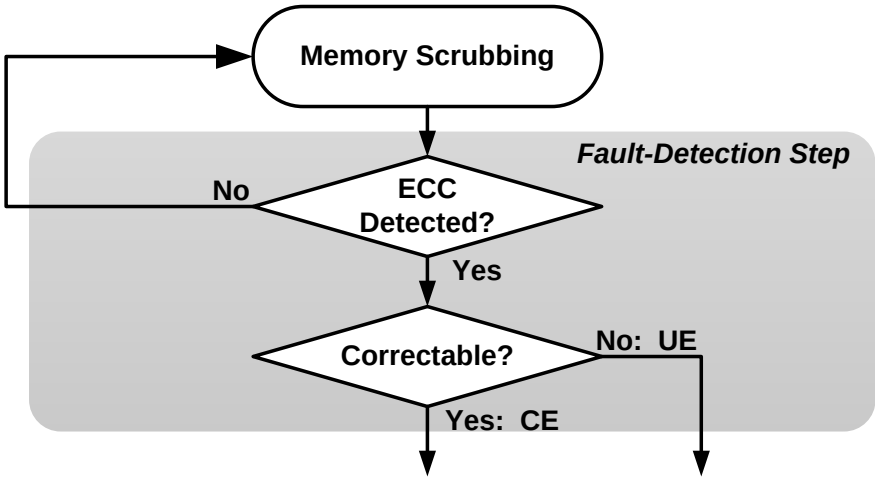


Figure 55 The flow chart of the proposed fault-detection scheme during memory scrubbing.

4.2.3.2 Fault Identification Using Sequences of Commands

To identify if an error detected and corrected by ECCs is a hard or soft error, we introduce a fault-identification method that exploits sequences of commands—*Read-Invert-Write-Read-Compare* or *Read-Write-Read-Compare*—shown in Figure 56. A complement/re-complement scheme developed for correcting double-bit errors, soft errors and hard errors, in a word with SECDEC ECCs [105] motivates the proposed combinations of commands. The main idea of the scheme is based on the fact that writing data back to an erroneous word fixes an intermittent error, such as a soft error or an interfacial error resulting from poor signal integrity in I/O circuitry, but does not fix a permanent error, such as a stuck-at aging fault.

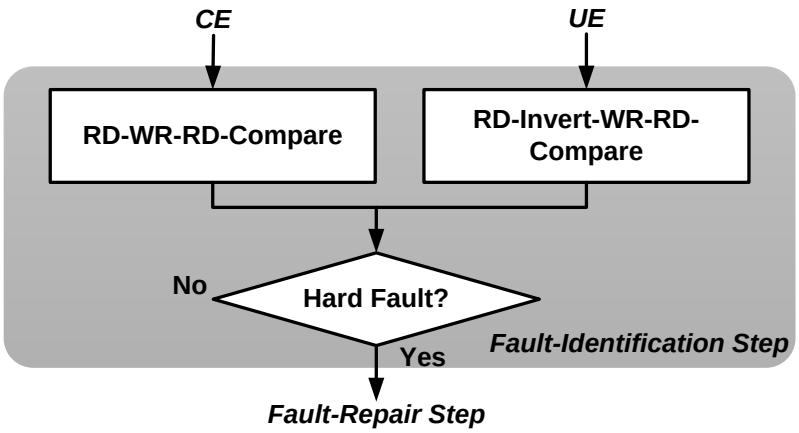


Figure 56 The flow chart of the proposed fault-identification scheme: *Read-Write-Read-Compare* for correctable errors (CEs) and *Read-Invert-Write-Read-Compare* for uncorrectable errors (UEs). RD = read. WR = write.

Since DRAM employs a double-data rate, data to/from a memory with a write/read command are accessed in the form of bursts. The length of a burst can be defined using the mode register set (MRS) as four or eight in the case DDR3 synchronous dynamic random access memory (SDRAM) [111]. In an ECC-DIMM, the ECC chip stores SECDED ECCs that are eight bits because a memory controller calculates ECCs for 72 bits, including 64

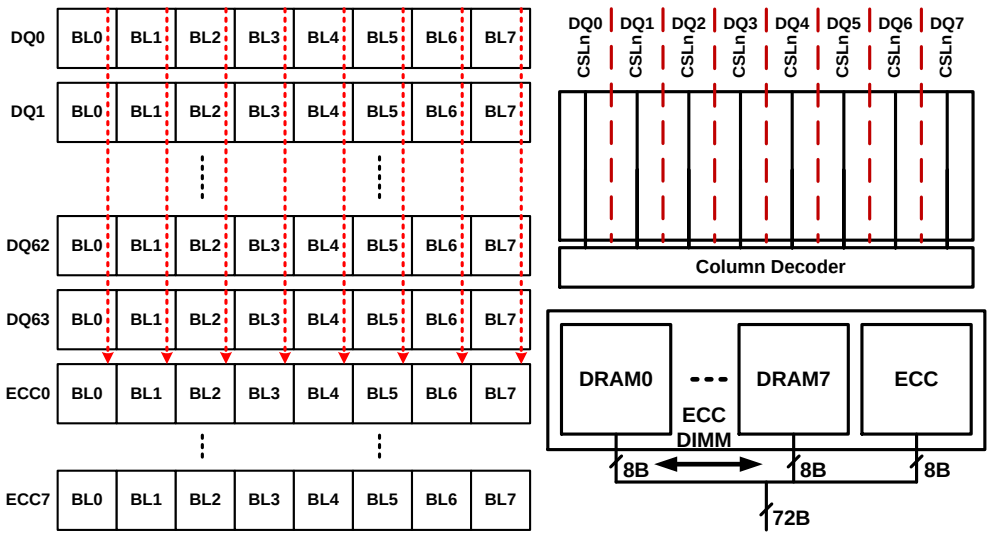


Figure 57 (a) A distributed word and its ECCs in an ECC-DIMM and (b) concurrently accessed column-selection lines for multi-bit prefetch (top) and the configuration of an ECC-DIMM (bottom) [101]. BL=burst.

DQs (data pins in the DDR circuitry), from eight DRAM chips and eight ECC bits. Therefore, as illustrated in Figure 57(a), each eight-bit ECCs, one-bit ECC for each eight-byte word, is calculated and stored on every burst. Since an ECC-DIMM contains one DRAM chip for ECCs and eight DRAM chips for data, depicted in the bottom of Figure 57(b), the memory controller accesses a DIMM with the granularity of a word. As illustrated in the top of Figure 57(b), eight column-selection lines (CSLs) are activated simultaneously in a DRAM chip with addresses from the memory controller. For an eight-bit prefetch and eight DQs, eight CSLs, each of which is connected to eight memory cells, are activated with a set of addresses, and 64 bits of data are read or written simultaneously in a bank of a memory. In a DIMM, a memory controller accesses 64-byte data with a set of addresses. Therefore, because of the large granularity of replacement (e.g., if a row is replaced based on the fail addresses, a DIMM wastes a page or several kilobytes), repairing faulty cells based on the fail addresses of a word is wasteful. To achieve finer granularity of repair, a DRAM chip should identify if an error in a word occurs with single-bit

granularity. Our proposed sequences of commands identify not only types but also positions of errors with single-bit granularity.

4.2.3.2.1 Read-Invert-Write-Read-Compare

Figure 58 and Figure 59 demonstrate how the proposed scheme identifies the type and the location of an error. We use an example with eight data bits in a burst ($b_0b_1b_2b_3b_4b_5b_6b_7$). Figure 58 illustrates cases of correctable errors. In the figure, b_i is the original data bit, and $\overline{b_i}$ is the inverted value of b_i. Since an inverted write operation alters a bit impacted by a soft error, not a hard one, '1' in the exclusive-or (XOR) results indicates the location of the hard error. In Figure 59, in the same context, two hard faults lead to two '1's in the comparison results, while one soft error and one hard error result in only one '1' in the results. For UEs, the number of '1's in the XOR results indicate the number of hard errors, as summarized in Table 3.

$b_0\ b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$	**Original data**
h	**Hard fault (*h*)**
$b_0\ \overline{b_1}\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$	**R0: Read erroneous data**
$\overline{b_0}\ b_1\ \overline{b_2}\ \overline{b_3}\ \overline{b_4}\ \overline{b_5}\ \overline{b_6}\ \overline{b_7}$	**W1: Write with inverted data**
$\overline{b_0}\ \overline{b_1}\ \overline{b_2}\ \overline{b_3}\ \overline{b_4}\ \overline{b_5}\ \overline{b_6}\ \overline{b_7}$	**R1: Read data after W1**
0 1 0 0 0 0 0 0	**W1 ⊕ R1: Compared results**

(a)

$b_0\ b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$	**Original data**
s	**Soft fault (*s*)**
$b_0\ \overline{b_1}\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$	**R0: Read erroneous data**
$\overline{b_0}\ b_1\ \overline{b_2}\ \overline{b_3}\ \overline{b_4}\ \overline{b_5}\ \overline{b_6}\ \overline{b_7}$	**W1: Write with inverted data**
$\overline{b_0}\ b_1\ \overline{b_2}\ \overline{b_3}\ \overline{b_4}\ \overline{b_5}\ \overline{b_6}\ \overline{b_7}$	**R1: Read data after W1**
0 0 0 0 0 0 0 0	**W1 ⊕ R1: Compared results**

(b)

Figure 58 Fault identification using *Read-Invert-Write-Read-Compare* in cases with a correctable error: (a) one hard fault and (b) one soft fault.

Table 3 Classifications of Errors During Field Operations

Types of Errors	Number of '1's in the Results	Correctable or Repairable
One soft	0	Correctable
One hard	1	Repairable/Correctable
Two hard	2	Repairable
One soft and one hard	1	Soft: Correctable after repair Hard: Repairable

$b_0\ b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$ **Original data**
$h \qquad\qquad h$ **Hard faults (*h*)**
$b_0\ \overline{b_1}\ b_2\ b_3\ b_4\ b_5\ \overline{b_6}\ b_7$ **R0: Read erroneous data**
$\overline{b_0}\ b_1\ \overline{b_2}\ \overline{b_3}\ \overline{b_4}\ \overline{b_5}\ b_6\ \overline{b_7}$ **W1: Write with inverted data**
$\overline{b_0}\ \overline{b_1}\ \overline{b_2}\ \overline{b_3}\ \overline{b_4}\ \overline{b_5}\ \overline{b_6}\ \overline{b_7}$ **R1: Read data after W1**
0 1 0 0 0 0 1 0 **W1 ⊕ R1: Compared results**

(a)

$b_0\ b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$ **Original data**
$s \qquad\qquad h$ **Soft fault (*h*) and hard fault (*s*)**
$b_0\ \overline{b_1}\ b_2\ b_3\ b_4\ b_5\ \overline{b_6}\ b_7$ **R0: Read erroneous data**
$\overline{b_0}\ b_1\ \overline{b_2}\ \overline{b_3}\ \overline{b_4}\ \overline{b_5}\ b_6\ \overline{b_7}$ **W1: Write with inverted data**
$\overline{b_0}\ b_1\ \overline{b_2}\ \overline{b_3}\ \overline{b_4}\ \overline{b_5}\ \overline{b_6}\ \overline{b_7}$ **R1: Read data after W1**
0 0 0 0 0 0 1 0 **W1 ⊕ R1: Compared results**

(b)

$b_0\ b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$ **Original data**
$s \qquad\qquad s$ **Soft faults (*s*)**
$b_0\ \overline{b_1}\ b_2\ b_3\ b_4\ b_5\ \overline{b_6}\ b_7$ **R0: Read erroneous data**
$\overline{b_0}\ b_1\ \overline{b_2}\ \overline{b_3}\ \overline{b_4}\ \overline{b_5}\ b_6\ \overline{b_7}$ **W1: Write with inverted data**
$\overline{b_0}\ b_1\ \overline{b_2}\ \overline{b_3}\ \overline{b_4}\ \overline{b_5}\ \overline{b_6}\ \overline{b_7}$ **R1: Read data after W1**
0 0 0 0 0 0 0 0 **W1 ⊕ R1: Compared results**

(c)

Figure 59 Fault identification using *Read-Invert-Write-Read-Compare* in cases with uncorrectable errors: (a) two hard faults, (b) one soft and one hard fault, and (c) two soft faults. '1's in the XOR results represent the error locations, and the number of '1's in the results indicates the number of hard faults.

4.2.3.2.2 Read-Write-Read-Compare

As described in Figure 60, for correctable errors, a sequence of commands without inverting, *Read-Write-Read-Compare*, is sufficient to identify the type and the location of a fault. Since SECDED ECCs correct a single-bit error in a word after reading a faulty word, a write command to the same address returns corrected data to a memory. As illustrated in Figure 60(a), comparing written data corrected by ECCs with read data followed by a write operation with correction produces '1' in the XOR results if a hard fault still remains in the word. Since writing corrected data to a faulty word that contains a soft error eliminates the soft error, the faulty word returns correct data after writing corrected data back into the memory, resulting in all zeros in the comparison results, as shown in Figure 60(b).

$b_0\ b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$	**Original data**
h	**Hard fault (*h*)**
$b_0\ \overline{b_1}\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$	**R0: Read data with a single-bit error**
$b_0\ b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$	**W1: Written corrected data**
$b_0\ \overline{b_1}\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$	**R1: Read data with a hard fault**
0 1 0 0 0 0 0 0	**W1 ⊕ R1: Compared results**

(a)

$b_0\ b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$	**Original data**
s	**Soft fault (*s*)**
$b_0\ \overline{b_1}\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$	**R0: Read data with a single-bit error**
$b_0\ b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$	**W1: Written corrected data**
$b_0\ b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7$	**R1: Read corrected data**
0 0 0 0 0 0 0 0	**W1 ⊕ R1: Compared results**

(b)

Figure 60 Fault identification using *Read-Write-Read-Compare* in cases with a correctable error: (a) a hard fault and (b) a soft fault. '1's in the XOR results represent error locations.

4.2.3.2.3 Proposed fault identification

For fault identification, we exploit *Read-Write-Read-Compare* for correctable errors and *Read-Invert-Write-Read-Compare* for uncorrectable errors, as shown in Figure 56. This is because after the fault identification step, unless an error is a hard fault, a memory system should be able to resume normal operations without degradation in performance. Although the command set *Read-Invert-Write-Read-Compare* successfully identifies the type and the location of both uncorrectable errors and correctable errors, such an identification step ruins the data integrity of a word by writing inverted data into the faulty word, unless the memory controller writes re-inverted data to the same word at the end of the fault identification step. For maintaining the data fidelity of a word containing a soft error, even after the fault identification step, and returning to normal operations with minimum degradation in performance during field operations, a memory system should avoid a command set with inversion as the fault identification method for correctable errors. Therefore, the proposed fault-identification scheme exploits the command combinations of *Read-Write-Read-Compare* for CEs and *Read-Invert-Write-Read-Compare* for UEs.

Note that it is possible that soft errors could occur during the fault identification step. However, ECCs can detect a soft fault in the second read operation (R1). Instead of comparing data from the write operation (W1) and R1, because a new correctable error in found, the memory controller starts a new fault-identification step. For the *Read-Write-Read-Compare* command sequence, the memory controller writes the inverted data of R1 again to the fault location of the memory (W2), reads them back (R2), and compares data from W2 and R2 to check if a fault is a hard or soft fault. Such a renewal of *Read-Write-Read-Compare* when a new error occurs during a fault-identification step successfully

identifies the type and location of a fault even if there are soft errors in the fault-identification step.

4.2.3.2.4 Circuit implementation of hard-fault detection

For the implementation of the fault detection (FD) scheme, if the DRAM retains the address in address latches and stores the latest read data in the read first-in-first-out logic, by simply redirecting the inverted read data to the write path, reading written data from the memory, and comparing written and read data, the type and the location of a fault is identified. However, given the variety of internal architectures from various memory vendors, we assume a worst case that the DRAM loses the latest read data and their address information. As illustrated in Figure 61, when a memory controller corrects an error in a word, it reads data from the memory, corrects the data using ECCs, rewrites them to the memory with the same address, and rereads them for comparison of written and read data. If the memory controller detects uncorrectable errors in a word, it reads the data again, inverts the data, and writes them back to the same memory cells, which is followed by a read operation from the same memory location. Comparing the written data and the second read data using exclusive-or logic indicates the type of error, that is, a permanent or an intermittent fault, and the location of a stuck-at-fault.

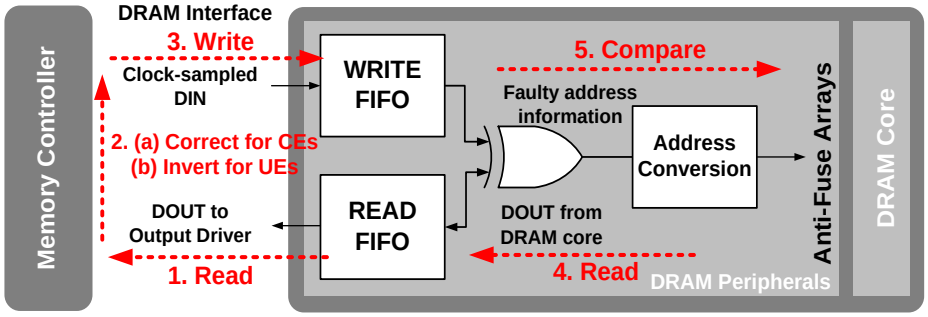


Figure 61 Circuit implementation for finding error sites in DRAMs. A memory controller writes data to a DRAM after it (a) corrects a CE and (b) inverts a UE.

Figure 61 also depicts the block diagram for the circuit implementation in a DRAM. The DRAM stores or latches data for the first read operation from an erroneous word as inputs to the read FIFO circuit. Then the write FIFO circuit stores or latches de-serialized written data from the write-back operation from the memory controller as outputs from the write FIFO. XOR circuits have been implemented for a comparison of read and written data. Since such identification logic is implemented in every DRAM chip, each DRAM chip checks if an error occurs in the memory. When an error is found in a DRAM chip, it stores the fail addresses in anti-fuse arrays for PPR. Since the data path of DRAMs is exploited, small modifications are needed for the implementation of the proposed scheme, including XOR gates for all DQs and bursts and multiplexers at the inputs to anti-fuse arrays.

4.2.3.3 Fault Repair Using Anti-Fuse Arrays in DRAMs

After detecting and identifying an error, each DRAM chip stores their fail address in available anti-fuses for remapping during PPR. To find a vacant fuse, a DRAM chip searches for a non-programmed master fuse that indicates that a fuse box is available. The method for the DRAM chip to search for an available master fuse defines the redundancy algorithm (RA). In this study, since a row-preferred repair scheme for manufacturing-level repair is assumed, the DRAM chip searches for an available master fuse for row redundancies first. If the DRAM chip finds an available fuse, it stops searching and stores the fail address in the available fuse. Otherwise, the DRAM chip continues to search for an available master fuse from among the column redundancies. If the DRAM chip finally finds an available fuse and successfully stores the fail address in the fuse, it sends a signal

indicating success to the memory controller to signal the need to return to normal operations. Otherwise, the DRAM chip sends a failure signal to the memory controller.

Figure 62 summarizes the overall flow of the post-package repair scheme during both in-field normal operations and memory scrubbing. For correctable errors, fault repair is performed only when a hard fault in a word is detected. After ECCs correct soft errors, the proposed scheme returns to original operations. For uncorrectable errors, three possible cases exist. For a case with two hard faults in a word, both bits are repaired. For a case with one soft and one hard fault in a word, since the soft error rarely recurs in the same location, only the hard fault is repaired. However, for a case with two soft errors in a word, since such errors rarely occur and disappear after reboot, there is no need to repair them.

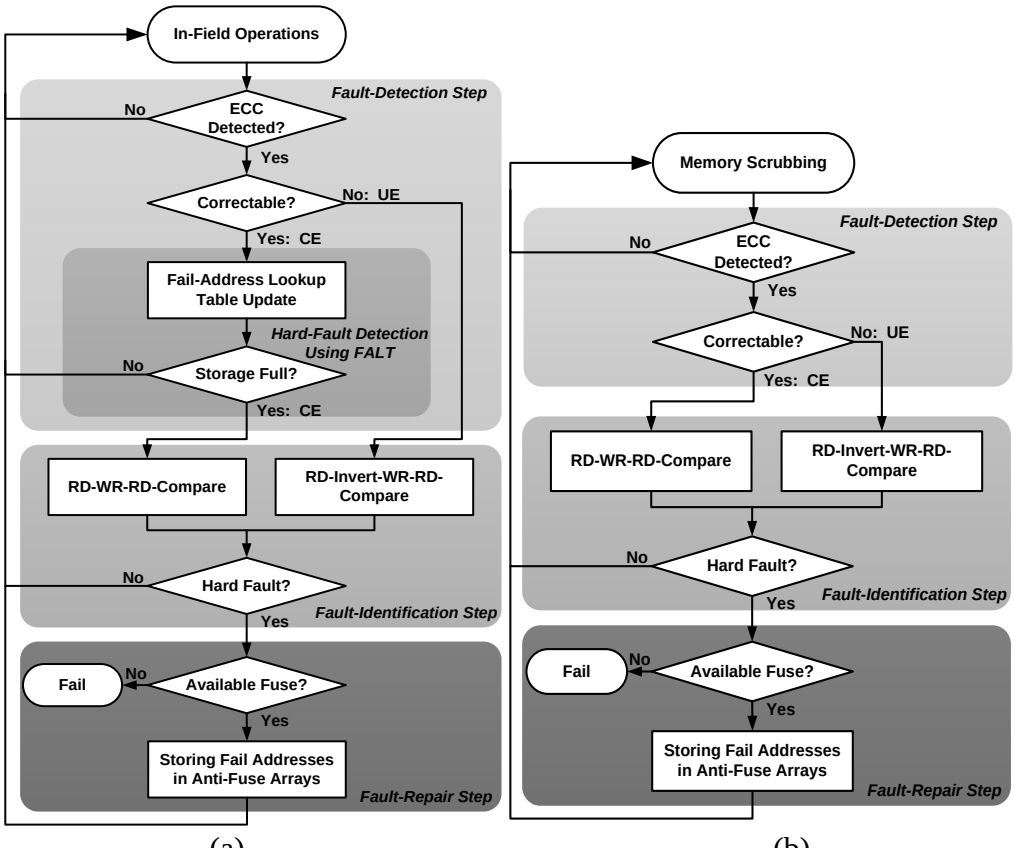


Figure 62 The flow charts of the proposed post-package repair scheme during (a) in-field normal operations and (b) memory scrubbing. RD = read. WR = write.

4.2.4 Experimental Setup

To evaluate the performance of the proposed repair scheme, we employ the memory yield and lifetime estimation methodology under aging errors proposed in prior work [107]. Since the prior work takes no manufacturing-level errors into account, we have implemented wafer-level yield simulation based on the methodology in [107].

For wafer-level errors, in our case study, we assume 15 % faulty rows, 10 % faulty columns, 5 % cluster faults, and 70 % single-bit faults [106]. We model aging errors based on in-field experimental data with an error rate of 25,000 to 75,000 FITs per Mbit in a DIMM [101]. For both manufacturing and aging characteristics, errors are distributed randomly in the memory.

We have implemented a redundancy analysis algorithm with conventional spare rows and columns[108], exploiting essential spare pivoting [109], which repairs faulty rows and columns that contain more than one fault in a row or a column first and then repairs single-bit faults. We also exploit a row-preferred repair algorithm that prioritizes the replacement of an error using a row redundant resource rather than a redundant column resource [110]. In our case study, we consider eight 2 Gbit DDR3 SDRAMs in a 2 Gbyte ECC-DIMM with 16 banks, 14-bit row addresses, and 7-bit column addresses [111].

4.2.5 Results and Discussion

Using a yield and reliability estimation methodology escribed in the previous section, we simulate a manufacturing-level repair scheme and the proposed scheme. In chapter 4.2.5.1, we investigate how many manufacturing-level redundancies remain after

wafer-level repair in a DRAM with wafer-level row/column redundancies. Chapter 4.2.5.2 examines the impact of the proposed post-package repair scheme on the improvement of DRAM lifetime.

4.2.5.1 Remaining Wafer-Level Redundancy

Using over one million Monte Carlo simulations, we estimated the yield of DDR3 SDRAMs with manufacturing/aging errors and various combinations of redundant resources. By varying the number and the combination of spare rows and columns for repairing wafer-level errors, we simulate the yield of a memory with 50 errors, which is 0.023 ppm per DRAM chip and a ten times higher defect density than in [106]. The results are shown in Figure 63. Regardless of the row/column redundancy combinations, since most errors are single-bit errors, the total number of redundant resources determines the yield of the example DRAM. With 14 redundant resources per bank, the yield reaches one.

Figure 63(b) and Figure 63(c) depict the probability of row and column redundancies remaining per bank, respectively. An average of 2.56 row and 0.22 column redundancies are used per bank. Because we exploit row-preferred repair, the probability of using redundant row resources is higher than that of redundant column resources. Since a prior study [112] employed 32 row redundancies per bank, in this case study we have used 32 spare rows and 16 spare columns per bank. In the case of a spare column connected to eight bitlines, to repair a CSL, we use a redundant column, which wastes eight times as many DRAM cells as a redundant row and requires more area for redundant cells than a spare row, because a spare column requires more redundant cell area. Therefore, we have assumed a smaller number of redundant columns are available. The efficiency of redundant

columns can be improved with the segmented column redundancy scheme. Our simulation results show that an average of 45.21 wafer-level redundant resources per bank are available for future use as redundancies for PPR after wafer-level repair.

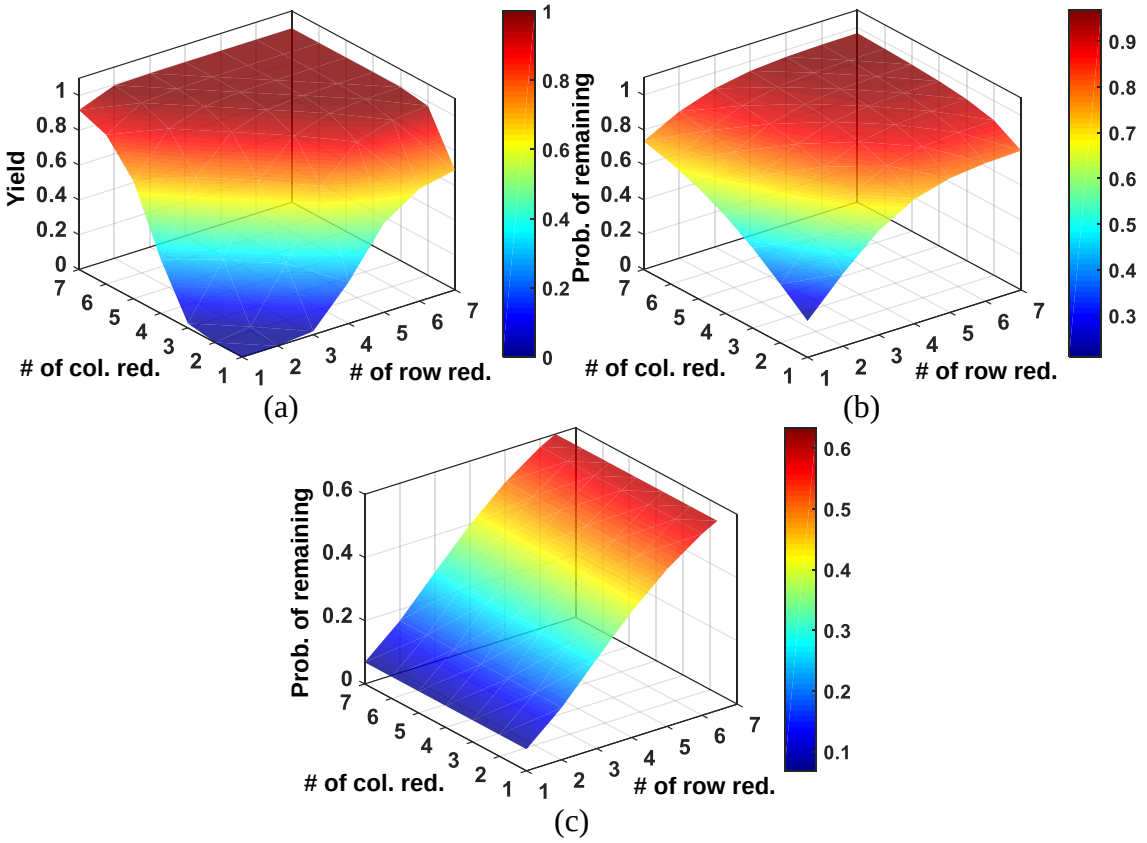


Figure 63 Simulated results of wafer-level repair with various numbers and combinations of redundant resources per bank in a 2 Gbyte DDR3 SDRAM (50 errors including 35 single-bit errors, three faulty columns, two faulty rows, and one four-bit cluster error): (a) the yield and probability of the remaining (b) column and (c) row redundant resources after wafer-level repair [101].

Figure 64 shows required numbers of redundancies per bank at a failure rate of 5,000 FITs per Mbit in a 2Gbit DDR3 SDRAM for ensuring various probabilities of a chip failure. Once the wafer-level defect density of a process technology node is characterized, the remaining numbers of redundancies after wafer-level repair is estimated. Since these remaining redundancies will be used to repair wearout failures, it becomes possible to determine the number of redundant resources required for a given yield and lifetime

requirement. Such an estimation gives a design guideline for circuit design engineers to optimize a repair scheme in a memory system.

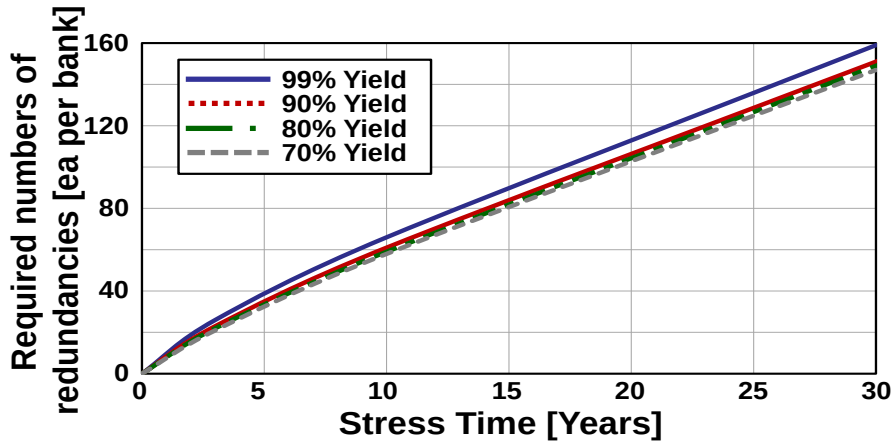


Figure 64 Simulated required numbers of redundancies per bank at a failure rate of 5,000 FITs per Mbit in a 2 Gbit DDR3 SDRAM with various stress times for achieving various yield values.

4.2.5.2 Analytic Estimation of Yield and Probability of Failure

This work has relied on yield estimates based on a Monte Carlo simulator [107]. However, in this section we determine analytic equations for the purpose of understanding the impact of redundancy and the probability of survival.

Assuming that errors are randomly distributed in a memory, we calculate the probability that a bank has no failures using the binomial distribution as follows:

$$P_{no\_fail,bank} = \sum_{k=0}^{N_{red}} P_k, \tag{35}$$

where

$$P_k = \binom{N_{bit}}{k} {F_{bit}}^k (1 - F_{bit})^{N_{bit}-k}, \tag{36}$$

and N_{red} and N_{bit} denote the number of redundancies in a bank and number of bits in a DRAM chip, respectively. The bit failure rate (F_{bit}) is N_{error}/N_{bit}, where N_{error} is the number of errors in a bank. From (35), the probability that a bank fails when the number of errors in a bank is higher than the total number of redundancies in a bank can be calculated as

$$P_{fail,bank} = 1 - P_{no\_fail,bank} = 1 - \sum_{k=0}^{N_{red}} P_k. \quad (37)$$

The yield of a memory is the probability that at least one bank fails within a memory.

$$Y_{memory} = {P_{no\_fail,bank}}^{N_{bank}}. \quad (38)$$

We propose a method that converts redundant resources for manufacturing errors into resources for aging errors. The analytic model derived above is still valid for reconfigured row/column redundancies for PPR. The only difference is the available number of redundancies. We define the initial number of redundant resources, $N_{red,0}$, as $N_{red,0} = N_{red,col} + N_{red,row}$, where $N_{red,col}$ and $N_{red,row}$ are the number of column and row redundancies, respectively, originally designed for manufacturing-level repair. After manufacturing-level repair, the number of remaining redundant resources available for PPR, $N_{red,rem}$, is as follows: $N_{red,rem} = N_{red,0} - N_{red,used}$, where $N_{red,used} = N_{red,col\_used} + N_{red,row\_used}$, and $N_{red,used}$, N_{red,col_used}, and N_{red,row_used} are the number of total, column, and row redundancies used for manufacturing repair, respectively. After applying N_{red} to (37), the probability of a bank failure for PPR is expressed as

$$P_{f,bank} = 1 - \sum_{k=0}^{N_{red,rem}} P_k. \quad (39)$$

To validate our simulation results, for various defect densities, we compare the simulated memory yield from Monte Carlo analysis [107] with the estimated yield based on (38) in Figure 65. The simulated yield closely corresponds to the calculated yield. A possibility that a redundancy replaces multiple errors in a row or a column causes a discrepancy between the simulation results and the calculation.

The probability that there are *k* defects is given by P_k in (36). In (35), if $k > N_{red}$, then it is assumed that the memory bank has insufficient resources to repair $p = k - N_{red}$ defects.

However, if all *p* defects fall in rows and columns that are already being repaired with redundant resources, then there will be no yield loss. The probability of falling on a row or column that is already being repaired is $N_{red-bit}/N_{bit}$, where $N_{red-bit}$ is the total number of redundant bits in the array. Hence, the yield of a bank is

$$P_{no\_fail,bank} = \sum_{k=0}^{N_{red}} P_k + \sum_{k=N_{red}+1}^{N_{bit}} P_k \left(\frac{N_{red-bit}}{N_{bit}}\right)^{k-N_{red}}. \quad (40)$$

Since the calculation/estimation of the number of redundancies used for repairing multiple bits is impractical, in this work, we exploit Monte Carlo yield simulation which accounts for the repairing of multiple errors with a single redundancy.

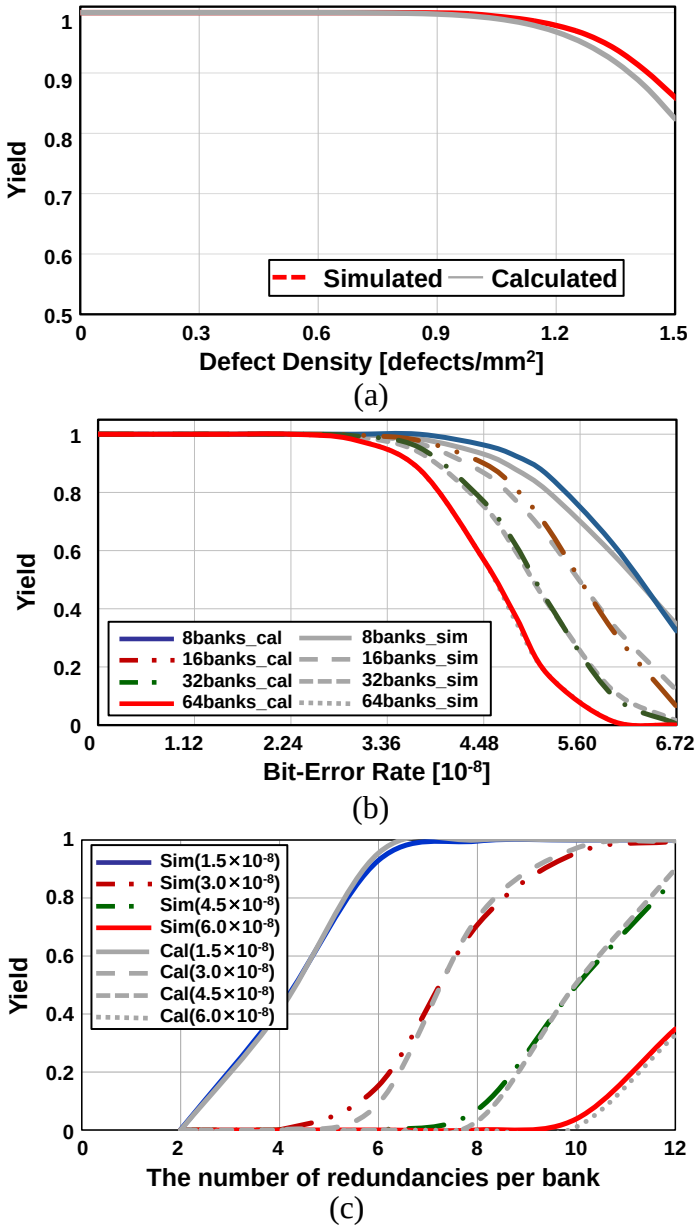


Figure 65 Simulated and calculated yield of the 2 Gbit DDR3 SDRAM with six-row and six-column redundancies per bank: (a) for various wafer-level defect densities with 16 banks, (b) for bit-error rates with various numbers of banks, and (c) for various numbers of redundancies per bank and bit-error rates.

4.2.5.3 Enhanced Reliability of a Memory System with the Proposed PPR Scheme

Figure 66 depicts the simulated results of memory yield with aging errors whose error rate is 75,000 FITs per Mbit in a DIMM. We define the lifetime of a DIMM as the time at which the probability of survival of a DIMM declines to 36.8 %, corresponding to

a 63.2 % probability of failure of a DIMM. We consider two cases, where memory scrubbing (via BIST and BISR) is used to identify failures at regularly scheduled downtime. The case with six spare rows and six spare columns is labeled as PPR, and the case with 12 spare rows and 12 spare columns is labeled as PPR+PPR_WF. The extra spare rows and columns extend the lifetime from 0.14 years to 0.68 years.

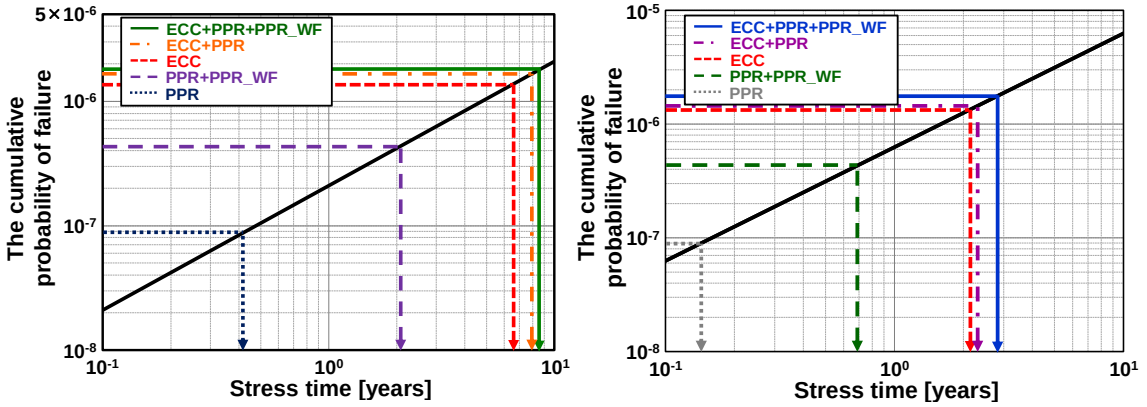


Figure 66 Simulated lifetime of a 2 Gbyte DDR3 ECC-DIMM with various combinations of ECCs and PPR for error rates of (a) 25,000 FITs per Mbit and (b) 75,000 FITs be Mbit (corresponding to 0.4 and 1.2 errors per hour, respectively). The case labeled PPR has six row and six column redundant resources per bank. The case labeled PPR+PPR_WF has six extra row and six extra column redundant resources per bank, since extra resources may be needed for in-field repair.

Using only ECCs, but no redundancies, the lifetime is 2.17 years. Combining ECCs and PPR with six spare rows and six spare columns extends the lifetime to 2.31 years. Adding an extra six spare rows and an extra six spare columns results in a lifetime of 2.85 years, which extends the lifetime of a memory over the standard practice of using redundancies to correct manufacturing errors and ECCs for in-field operations by 31.4 %. We have also considered four-segmented column redundancies for both wafer- and aging-level errors, depicted in Figure 67. With segmented column repair, the lifetime of PPR-only repair increases from 0.14 to 0.36 years. Combining PPR using wafer-level redundancies with ECCs and four-segmented PPR increases the lifetime from 2.71 to 3.28

years. Such results show that a segmented column redundancy scheme with fine granularity is efficient for in-field PPR.

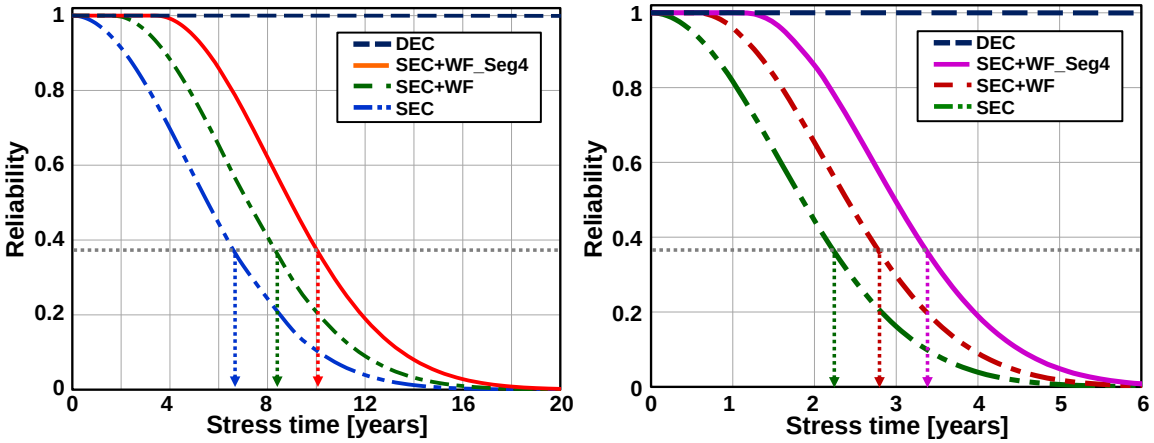


Figure 67 Simulated lifetime of a 2 Gbyte DDR3 ECC-DIMM with various combinations of ECCs and wafer-level redundancies for PPR (PPR_WF) for repairing aging errors with (a) 25,000 FITs and (b) 75,000 FITs per Mbit (corresponding to 0.4 and 1.2 errors per hour, respectively): with and without segmented column-redundant resources. Reliability is the probability of survival.

We compared stronger ECCs such as double-error correction and ternary-error detection (DECTED) that corrects double errors in a word with the proposed scheme. Figure 67 shows that DECTED drastically enhances the reliability of the memory. However, such stronger ECCs increase the latency of memory accesses, resulting from the delay of ender/decoder circuits, which not only degrades memory performance, but also requires modifications in a memory system.

4.2.5.4 Risk of System Failure

Since we exploit SECDED ECCs, a system fails when a double-bit error occurs in a word. If we define the number of words in a DIMM as N_{word}, we can obtain N_{word} by dividing the total number of bits in a DIMM, $N_{bits,DIMM}$, by the number of bits in a word, $N_{bits,word}$ as

$$N_{word} = \frac{N_{bits,DIMM}}{N_{bits,word}}. \quad (41)$$

For example, a 2 Gbyte DIMM with eight-byte words has $N_{word} = 2^{28}$ words. The probability of the failure of a DIMM is

$$P_{DIMM.fail} = 1 - (1 - P_{word})^{N_{word}} \approx N_{word} P_{word}, \quad (42)$$

where P_{word} is the probability of the failure of a word and P_{bit} is the probability of the failure of a bit. Since a word fails if there are two or more errors, we calculate P_{word} using the binomial distribution as

$$P_{word} = 1 - (1 - P_{bit})^{n_{bit}} - n_{bit} P_{bit} (1 - P_{bit})^{n_{bit}-1} \quad (43)$$

$$\approx n_{bit}(n_{bit} - 1) {P_{bit}}^2, \quad (44)$$

where n_{bit} is the number of bits in a word. Overall,

$$P_{DIMM.fail} \approx N_{word} n_{bit}(n_{bit} - 1) {P_{bit}}^2. \quad (45)$$

If there are N_{error} failed bits, then $P_{bit} = N_{error}/(N_{word} n_{bit})$. Thus,

$$P_{DIMM.fail} = 1 - ((1 - P_{bit})^{n_{bit}} + n_{bit} P_{bit} (1 - P_{bit})^{n_{bit}-1})^{N_{word}} \quad (46)$$

$$\approx \frac{{N_{error}}^2}{N_{word}}. \quad (47)$$

Let N_{PPR} be the FALT size that activates PPR. During an interval between PPRs, N_{PPR} faults accumulate. As a result, the probability of failure is

$$\Delta P_{DIMM.fail} \approx \frac{{N_{PPR}}^2}{N_{word}}. \quad (48)$$

This equation indicates that the size of the FALT determines the risk of system failure.

As an example, for a 12-hour interval, suppose that N_{PPR} faulty bits are stored. PPR repairs, at most 15 errors, which reduces the probability of the survival/reliability of a DIMM by 8.38×10^{-7}. Note that if the size of the FALT is 15, the failure rate is 75,000 FITs per Mbit, and we check the full memory every 12 hours, the risk of system failure for our PPR scheme is the same as the conventional BISR test which performs reconfiguration every 12-hour scheduled maintenance interval. However, our PPR scheme is event activated. This means that if the failure rate is lower or higher than expected, the risk of system failure between scheduled maintenance intervals is constant and controlled by design.

4.2.5.5 Overhead of the Proposed Scheme

4.2.5.5.1 Performance Overhead

Our proposed *Read-Invert-Write-Read-Compare* scheme incurs approximately 150 ns delay for three DRAM accesses, together with several gate delays of a few hundred picoseconds and the latency of a DRAM access, which is approximately 50 ns [113]. At a correctable failure rate of 75,000 FITs per Mbit, including soft errors [26], one correctable error occurs every 0.81 hours.

To mitigate soft errors, we perform memory scrubbing using ECC-DIMMs in data centers [113]-[116]. To compare overheads of the proposed scheme with those of memory

scrubbing, we assume a 12-hour scrub interval [114]. In the case of memory scrubbing with a 12-hour interval with a failure rate of 75,000 FITs per Mbit, a memory system is likely to have 14.8 correctable errors. Assuming that all errors are hard errors, our scheme takes 2,250 ns, which is 3.065×10^{-3} % of the scrub performance overhead of 73.4 ms (280 ns per row × 2^{18} rows) [113] for PPR during memory scrubbing, corresponding to 5.2×10^{-9} % of total available execution time for ECC-activated PPR.

For in-field operations, the performance overhead depends on the time needed to fill the FALT. In general, for a case with N_{PPR} correctable errors which are required to activate PPR, the proposed scheme takes N_{PPR}×150 ns. Since the incidence rate of uncorrectable errors is much lower than that of correctable errors [26]-[28], the performance degradation resulting from uncorrectable errors is negligible.

Unlike the latency of the proposed scheme, which is seldom incurred, i.e., only when the FALT is full, and the full FALT initiates a Fault Identification step, stronger ECCs of DECTED always increase the latency of a memory system, which may double the latency of a memory access [115].

4.2.5.5.2 Area Overhead

Since with a failure rate of 75,000 FITs per Mbit, the 12-hour interval scrubbing leads to an average of 14.8 correctable errors. We conservatively set a trigger value of the FALT as 01_2 and assume 15 entries of FALT. The storage overhead of the proposed scheme in a 4 MB LLC is 1.2×10^{-3} % with 15 entries of FALT, each of which contains a 25-bit fail address and a two-bit counter. For the circuit implementation, we require XOR gates for every DQ and a multiplexer for feeding a fail address to the anti-fuse arrays. Our 2 Gbit

DDR3 DRAM design with 20 nm technology [101] incurs area overheads that are less than 0.0016 % of total area of a chip.

Although stronger ECCs such as DECTED can improve the lifetime of a memory system, they require 15 check bits [115],[116], corresponding to 23.4 % area overheads for redundant cells for storing ECCs. DECTED ECCs require additional 10.9 % area overheads for check bits. In addition to check bits, implementing DECTED in a memory system requires modifications in encoders and decoders in a memory controller, which may require two times or more area than the area of encoders and decoders for SECDED [115]. Moreover, stronger ECCs necessitate changes in the architecture of ECC-DIMMs, which undermines the JEDEC standards of DRAMs.

4.2.6 Summary

To enhance the in-field reliability and to extend the lifetime of a memory system, we have proposed a post-package repair scheme. Unlike prior work using BIST and BISR, this work exploits ECCs and a memory controller. When a memory controller detects errors using ECCs, it temporarily stores the fail addresses of correctable errors in a fail-address lookup table in the last-level cache. To minimize degradation of system performance, when all entries in the fail-bit address table are full, the memory controller initiates fault identification with the sequence *Read-Write-Read-Compare* for correctable errors. If an error is corrected during memory scrubbing or if a double-bit error is detected, the memory controller initiates the fault identification stage immediately using the sequence, *Read-Invert-Write-Read-Compare,* for uncorrectable errors. After detecting and identifying types and locations of errors, if an error is a hard error, the fail address is stored in an

available anti-fuse of a DRAM for remapping. By eliminating correctable errors that potentially incur uncorrectable errors, our post-package repair enhances the lifetime of a memory system.

For validation, we have implemented a memory yield simulator that takes both manufacturing and aging errors/redundancies into account when estimating the yield. We demonstrated that our proposed PPR scheme extends the lifetime of a memory system by exploiting ECCs and PPR with small area and performance overheads. We have shown that the reuse of wafer-level redundancies for aging errors substantially extends the lifetime in comparison with an ECC-only scheme. We also demonstrated that a segmented column redundancy scheme is efficient for extending the lifetime of a memory system by more effectively using spare columns in the post-package repair of DRAMs.

CHAPTER 5. TEST AND DIAGNOSIS OF WEAROUT ERRORS IN MEMORY

5.1 Built-In Self-Test in Embedded DRAMs

5.1.1 Introduction

For better performance with less cache misses, the last-level cache requires high capacity. As an LLC, embedded dynamic random access memory (eDRAM) is attractive because it has much higher density than static random access memory (SRAM), widely used as cache memories. The small cell area causes eDRAM to be more vulnerable to reliability issues. Therefore, scaling the feature sizes of eDRAM, the path to higher density, also raises further reliability concerns.

Among the wearout mechanisms, frontend wearout resulting from bias temperature instability, hot-carrier injection, and gate oxide breakdown degrades the performances of transistors [40]-[42]. BTI and HCI increase the threshold voltages (V_{th}) of MOSFET devices. Such increases in V_{th} reduce the operating currents of devices and cause performance degradation of circuits. Meanwhile, GTDDB creates paths from the gate of a transistor to either its drain or source, also causing malfunctions and degradation of circuit performances.

In this work, we analyze the impact of frontend wearout mechanisms on eDRAM. To investigate the impact of BTI and HCI on eDRAM cells, we simulate eDRAM operations with the increased V_{th} of cell transistors using SPICE. To model GTDDB in circuit simulation, we introduce a gate-level model employing resistors that connect a

wordline to either a bitline or a storage node (SN), whose resistance decreases as the device ages. Based on observations from the analysis, we propose a proactive test algorithm for eDRAM that can be implemented as a built-in self-test circuit that monitors the degree of degradation resulting from frontend wearout in eDRAM. Using our test algorithm, we not only monitor the degradation but also distinguish GTDDB between a gate and a storage node from GTDDB between a gate and a bitline.

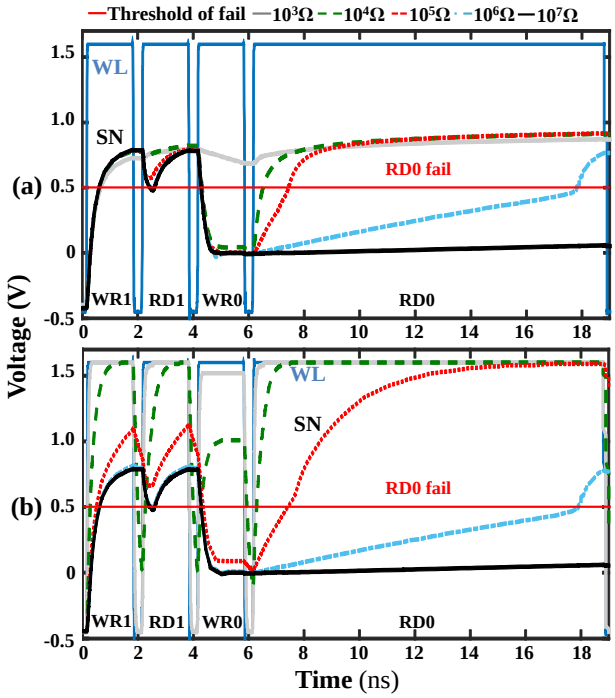


Figure 68 Simulation results of GTDDB (WR1-RD1-WR0-RD0): (a) GTDDB $_{\text{G-to-BL}}$ with long a RD0 and (b) GTDDB$_{\text{G-to-SN}}$ with a long RD0.

5.1.2 Detection of Frontend Wearout in eDRAM

To detect GTDDB in eDRAM, our analysis demonstrates that the resistance of paths from GTDDB should be smaller than $10^5\ \Omega$. However, note that at least one failure in a 32 MB LLC after two and half years of stress time is expected with the number of GTDDB paths corresponding to $10^6\ \Omega$ resistance, as shown in Figure 27. To enhance the sensitivity of our GTDDB detection scheme, we utilize read operations with a long duration.

A read operation as long as 13 ns can detect read '0' failures resulting from a 10^6 Ω resistance. The signals of the $GTDDB_{G\text{-}to\text{-}BL}$ fault, illustrated in Figure 68(a), show similar results to those of the $GTDDB_{G\text{-}to\text{-}SN}$ fault, as shown in Figure 68(b).

Because of leakage from the storage node with a high voltage level to a wordline precharged to a negative bias during the standby mode, a $GTDDB_{G\text{-}to\text{-}SN}$ fault, unlike a $GTDDB_{G\text{-}to\text{-}BL}$ fault, causes read '1' to fail with a long interval between write '1' and read '1' operations, as shown in Figure 69. Therefore, we can distinguish $GTDDB_{G\text{-}to\text{-}BL}$ from $GTDDB_{G\text{-}to\text{-}SN}$ using intervals between write '1' and read '1' operations.

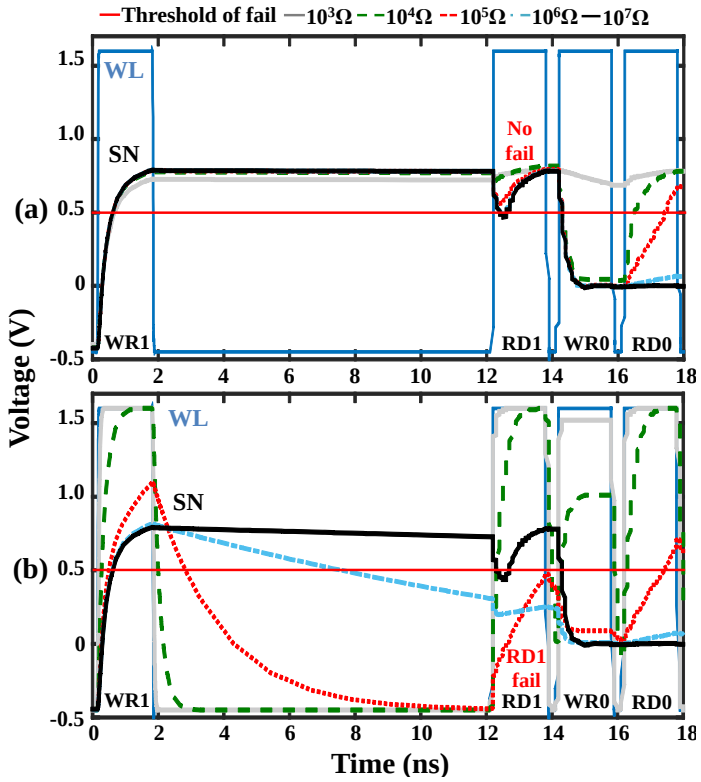


Figure 69 Simulation results of GTDDB (WR1-RD1-WR0-RD0): (a) $GTDDB_{G\text{-}to\text{-}BL}$ with interval between WR1 and RD1 and (b) $GTDDB_{G\text{-}to\text{-}SN}$, with an interval between WR1 and RD1.

Because current on a wordline (I_{pp}) during the write '0' operation increases as the resistance of GTDDB decreases (see Figure 70), we can monitor GTDDB with I_{pp}. Current flows from a gate (connected to an activated wordline) to either a bitline or a storage node

written as ‘0’ through the GTDDB resistor. This current increases as the resistance decreases. In the two GTDDB cases, when the resistance is under $10^5\,\Omega$, the difference in the current I_{pp} as a function of resistance results from the limited charge capacity of the cell capacitor for the $GTDDB_{G\text{-}to\text{-}BL\ case}$.

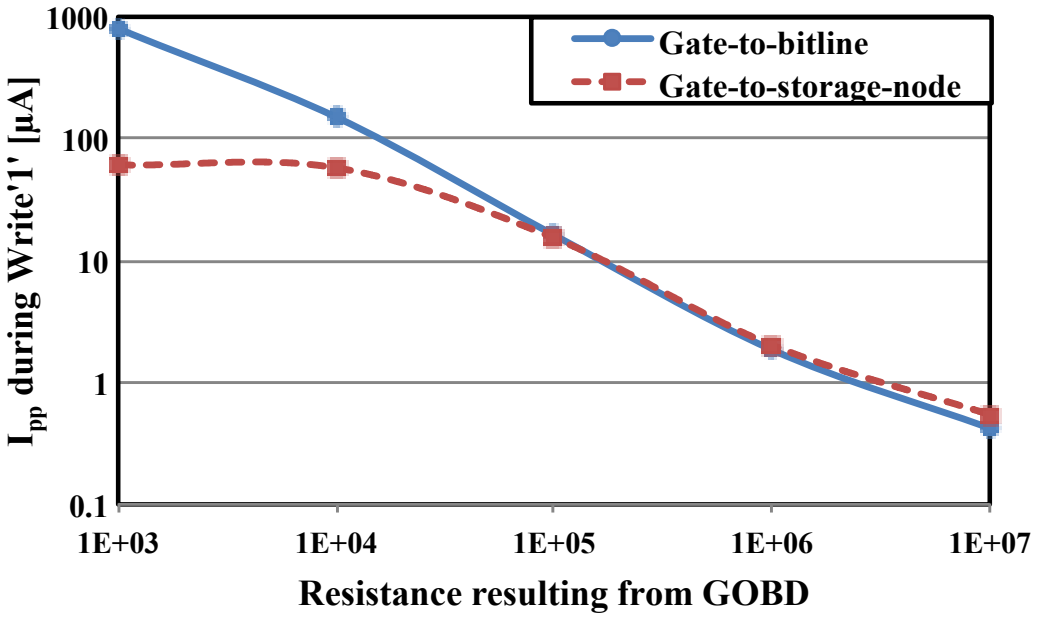


Figure 70 Simulated WL current during the write ‘0’ operation with different resistances.

5.1.3 Reliability Test Algorithm for eDRAM

To design BIST for detecting failures resulting from frontend wearout, note that the impact of BTI and HCI is negligible, and GTDDB is dominant, which is detected as summarized in Table 4. Since boosting the voltage level of a wordline enhances the driving current of a cell transistor in eDRAM with a high V_{gs}, a bias generator is used for the wordline bias (V_{pp}) [117]. By measuring the current, I_{pp}, we monitor GTDDB in the wordline direction using a current detector added to the generator, whose general implementation is shown in Figure 71. To detect GTDDB in the bitline direction, we can exploit the read ‘0’ operation with various read durations. By increasing the duration of the

read ‘0,’ the detectable resistance resulting from GTDDB increases. To differentiate $GTDDB_{G\text{-}to\text{-}BL}$ from $GTDDB_{G\text{-}to\text{-}SN}$, we use write ‘1’ and read ‘1’ operations at various intervals between write and read operations. A larger interval also increases the detectable resistance of $GTDDB_{G\text{-}to\text{-}SN}$.

Table 4 Test modes and test patterns for GTDDB monitoring

Mode	Test Point	Test Patterns
WL-direction	Vpp current	(w0)
BL-direction	Dataline Pass/Fail	(w0, r0 with increasing read time)
Differentiation	Dataline Pass/Fail	(w1, increasing interval, r1)

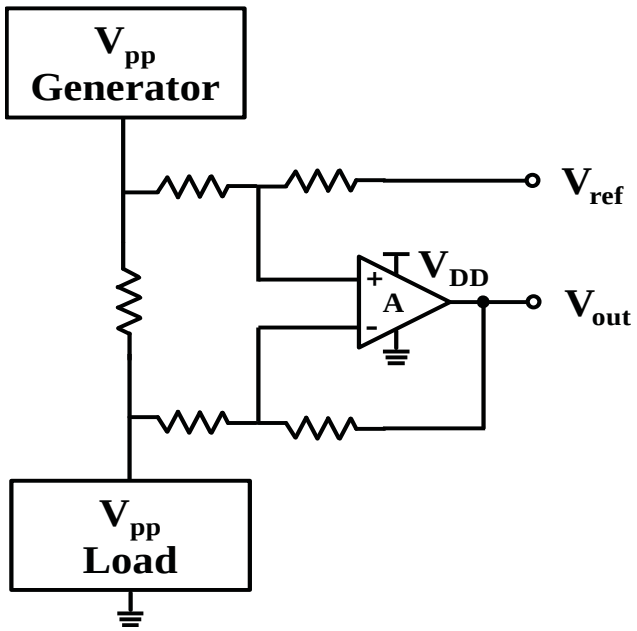


Figure 71 A current detector in the V_{pp} generator.

Figure 72 presents the algorithm for testing eDRAMs under the impact of frontend wearout. While writing ‘0’s to eDRAMs, we detect GTDDB in a wordline by monitoring I_{pp}. If we detect a difference larger than several μA in I_{pp}, we proceed to read the ‘0’ data to detect the read ‘0’ failures in the wordline to identify the position of the failure in the memory array. Since only $GTDDB_{G\text{-}to\text{-}SN}$ results in a read ‘1’ failure, testing of the write

‘1’ and read ‘1’ operations with an interval distinguishes $GTDDB_{G\text{-}to\text{-}BL}$ from $GTDDB_{G\text{-}to\text{-}SN}$ in the failed bit. To detect wearout in eDRAMs, the interval for monitoring GTDDB can initially be coarse (e.g., once every six months), but fine (e.g., once every one month) after a certain amount of time (e.g., one year), based on the expected probability of failure, as suggested in Figure 27, for a reduction of the total test time.

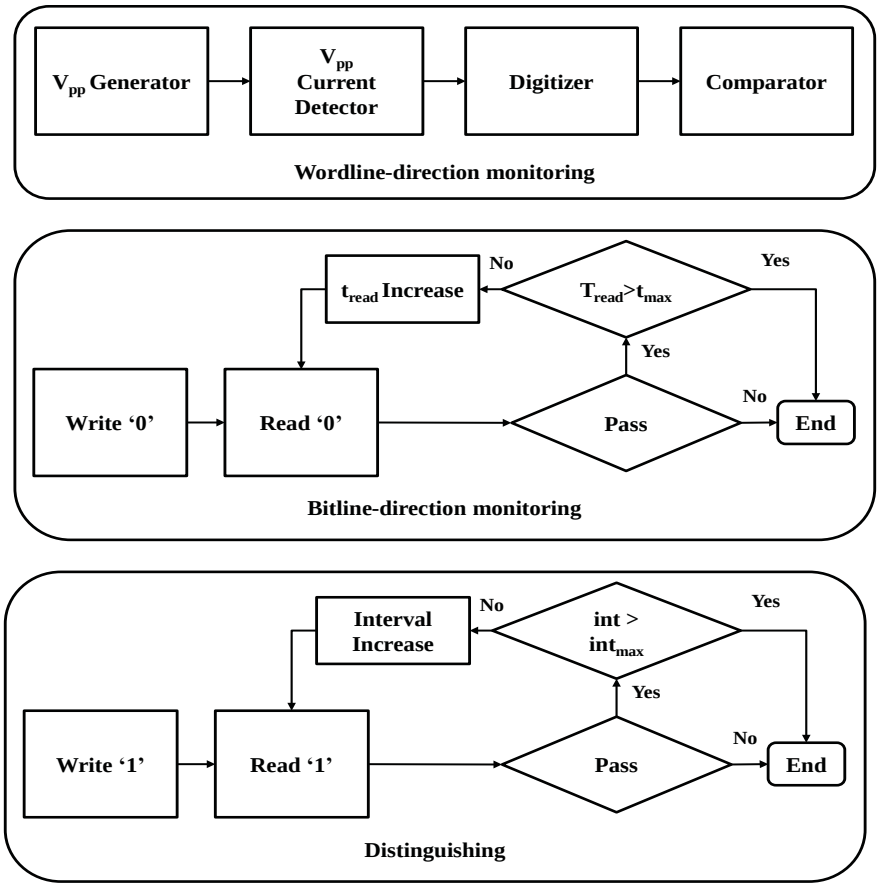


Figure 72 Test algorithm for detecting and diagnosing frontend wearout failures in eDRAMs.

5.1.4 Summary

This work has analyzed the impact of frontend wearout mechanisms, namely as BTI, HCI, and GTDDB, on eDRAMs. Since an eDRAM cell consists of one transistor and one capacitor, we have investigated the impact of these mechanisms on both the cell transistor and the cell capacitor.

Our simulation results have demonstrated that degradation of the transistor from BTI is negligible because of the robust micro sense amplifier with a high transfer ratio. The impact of BTI on the cell capacitor is also negligible because the gate bias of the cell capacitor is smaller than that of the cell transistor.

For HCI, since the cell transistor has a low drain voltage during write operations, degradation of the cell transistor resulting from HCI can be ignored. Since no drain voltage is applied to the cell capacitor, the impact of HCI on the capacitor is also negligible.

Because the impact of the gate voltage on the cell capacitor is much smaller than that on the cell transistor, the reduction in the lifetime of the cell transistor resulting from GTDDB is dominant in comparison with the cell capacitor. We have, therefore, investigated only GTDDB in the cell transistor and have ignored GTDDB in the cell capacitor.

As for GTDDB in the cell transistor, because it causes a stuck-at-'1' fault in the bitline direction during the write/read operations of data '0,' a read '0' operation fails. We detect a resistance of $10^5\ \Omega$ or smaller, and a 13 ns longer read operation increases the detectable range to $10^6\ \Omega$. With GTDDB, a current path is formed from a wordline to either a drain or a source.

This study has found that the current sensing of I_{PP} helps monitor and detect GTDDB in the wordline direction. With detection schemes for both the bitline and the wordline direction, one can determine the fault location. We also found that a test pattern of write '1' and read '1' with increasing intervals between operations helps to distinguish

if the fault comes from $GTDDB_{G\text{-}to\text{-}SN}$ or $GTDDB_{G\text{-}to\text{-}BL,}$ since only $GTDDB_{G\text{-}to\text{-}SN}$ results in a read '1' failure.

After investigating the impact of frontend wearout mechanisms on an eDRAM cell, we have proposed a built-in self-test algorithm for detection of frontend wearout mechanisms, to detect GTDDB in both the bitline and the wordline directions and to determine if GTDDB occurs between the gate and either its drain or source of a cell transistor.

5.2 System-Level Accelerated Life Test

5.2.1 Introduction

To estimate the lifetime of a circuit/system, degraded by wearout, device-level reliability tests are conventionally employed. G. Groeseneken et al. [118] argues that the classical approach to estimating reliability of a circuit/system by relying on reliability assessment at the technology level is inadequate for advanced CMOS technologies. In technology-level reliability assessment, we assume that the failure of a device results in a circuit/system failure. However, error-tolerance of a circuit/system to device-level wearout failures should be considered. Hence, circuit/system-level reliability assessment is necessary to accurately estimate lifetime of a circuit/system. Risk of in-field catastrophic failures is high if there is no circuit-level lifetime assessment to complement technology qualification at a device level.

Since most of the dielectric breakdown wearout failures come from memories within a processor [48], to assess the reliability of a computer system, we especially need

to find a memory reliability assessment technique. In this research, we assume that the reliability of a memory system mainly degrades because of TDDB. Using an open-source microprocessor, Leon3 [88], as a case study, we simulate the failure probability of the memory system resulting from each TDDB wearout mechanism. From the simulated probability of failure, we define acceptability regions for each TDDB mechanism for system-level accelerated life test. Acceptability regions indicate the best test conditions for a circuit/system, while taking the real operating environment of the circuit/system into account.

System-level accelerated life test typically involves stress and test sequences, where testing involves built-in self-test. We propose to detect failures throughout the stress period with error-correcting codes, unlike prior work. Therefore, we account for the ECCs scheme, single-bit error correction (SEC) and double-bit error correction (DEC), in selecting the optimal design of experiments for ALT.

5.2.2 Reliability Assessment Methods

5.2.2.1 The Classical Approach to Reliability Assessment

In the conventional reliability assessment, a device-level lifetime distribution is extracted of each specific wearout mechanism, while other wearout mechanisms are suppressed because the test structures are designed to be sensitive to each wearout mechanism [12],[43],[118], as shown in Figure 73(a). By using such dedicated test structures, we isolate failures resulting from each target wearout mechanism from those originating from others. From the empirical results from device-level accelerated life test

for each wearout mechanism with dedicated test structures, we extract lifetime data, including acceleration factors and wearout model parameters.

If the device-level lifetime distribution meets the expected probability of failure (P_{fail}) at a specified stress time, we statistically conclude that a system implemented using such reliability-qualified devices will work throughout the required lifetime of a system. Consequently, statistical estimation of system reliability demands a large reliability margin on devices to guarantee system-level reliability [118]. Moreover, such estimation of system reliability fails to reflect not only the real operating conditions but also configurations of the circuit/system that significantly influence the degradation of lifetime.

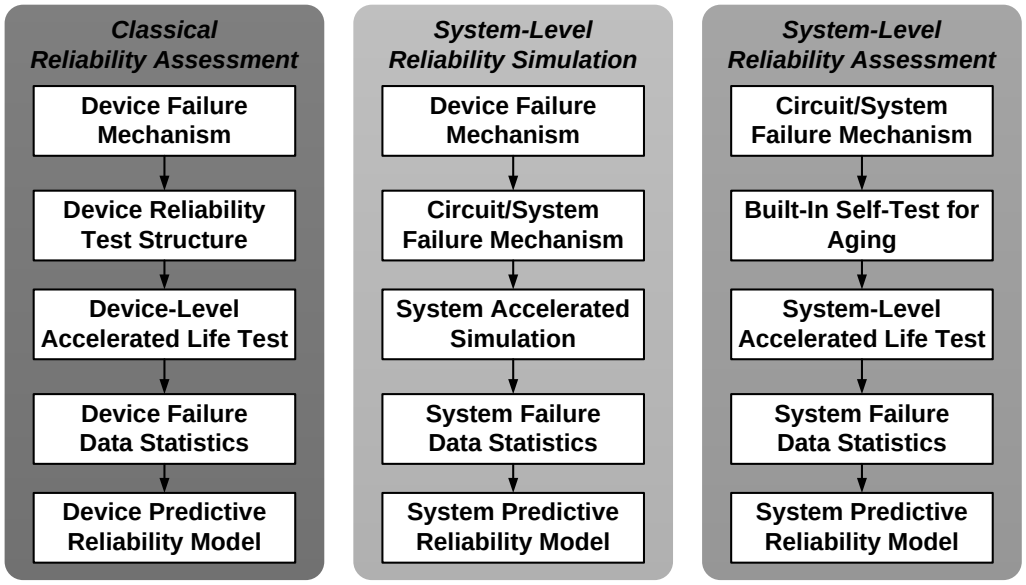


Figure 73 Various reliability assessment methods: (a) classical device-level assessment, (b) system-level reliability estimation using simulation, and (c) system-level assessment using a system-level accelerated life test.

5.2.2.2 Reliability Assessment Using Simulation

Recent research has demonstrated methodologies of estimating system-level reliability using simulation by taking the real operating environments of aging [44]-[46]. Since such studies account for benchmarks and circuit/system details and configurations,

the estimation of system-level reliability based on such work is more accurate than that of the conventional method based on device-level measurements.

In this research, we employ a reliability estimation method illustrated in Figure 73 (b). Since the calibration of simulations with empirical data at a system level is still on its way, our failure statistics at a system level in this study are based on failure data statistics from simulated experimental results generated by a system-level reliability simulation methodology [44] that is built on analytical models and calibrated from device-level testing results [5]-[7],[9]-[12],[44],[80],[96]-[124].

5.2.2.3 Proposed Methodology for Reliability Assessment

To complement the conventional device-level reliability testing for estimating circuit/system-level lifetime, we propose circuit/system-level reliability testing, as depicted in Figure 73(c). The challenge is that unlike device-level test with dedicated test structures that isolate target wearout failures from other wearout mechanisms, system-level test includes mixed failures simultaneously caused by various wearout mechanisms. To distinguish the cause of a failure among various wearout mechanisms, we aim to identify test conditions that isolate each wearout mechanism as much as possible, while ensuring that they are visible. In this way, we ensure coverage of all critical wearout mechanisms.

To do this, we rely on system-level reliability simulation that takes into account the real operating environment via realistic workloads [1],[4],[8],[11],[44]-[46]. In the simulation models, failure data statistics are based on analytical models and calibrated from device-level testing results [5]-[7],[9]-[12],[44],[80],[96]-[124]. The proposed work flow is illustrated in Figure 74. The aim of this work is to use reliability simulation under

realistic workloads to select optimal test conditions. Moreover, the ability to use ECCs to detect faults allows us to use stress scenarios that are more realistic and reflective of actual usage while collecting failure data.

Furthermore, the selected optimal test conditions can enable calibration of the system-level reliability simulator. It provides verification that device models are sufficient to characterize circuit/system reliability.

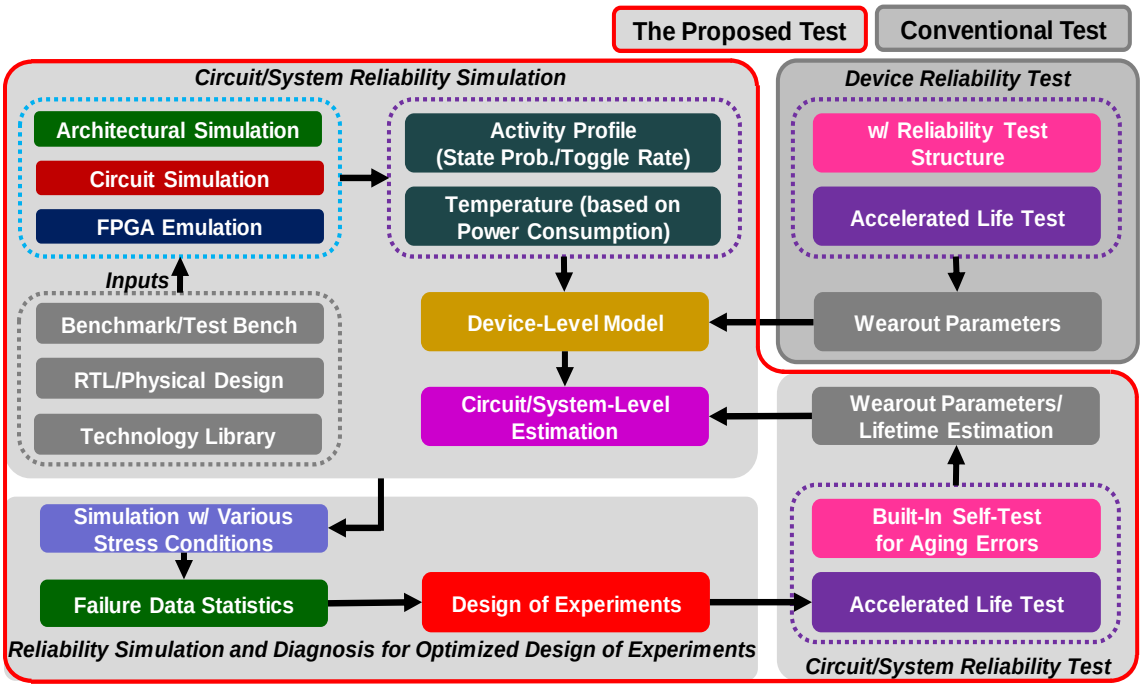


Figure 74 The proposed workflow of this research.

5.2.3 System-Level Accelerated Life Test

For system-level accelerated life tests, we can employ various test flows, as shown in Figure 75. All methods involve collecting samples of failure data, extracting wearout parameters, calculating acceleration factors, and estimating circuit/system reliability at use conditions, which combines the wearout parameters with the acceleration parameters. The differences relate to the method to apply stress and to collect failure data.

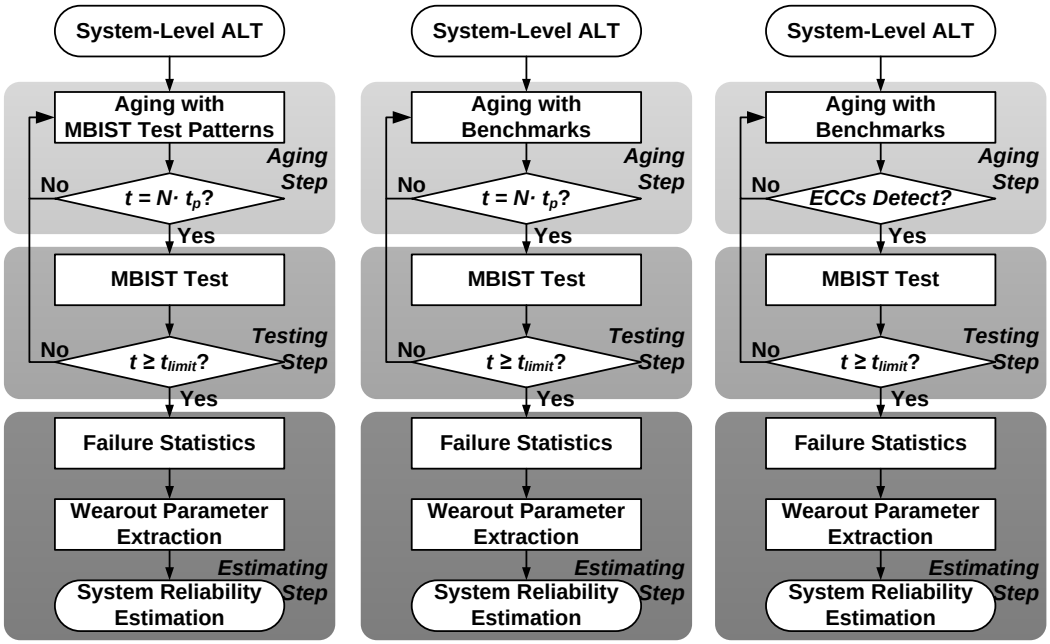


Figure 75 Various test flows for system-level accelerated life tests using BIST: (a) aging with test patterns and testing with BIST, (b) aging with benchmarks and testing with BIST, and (c) aging with benchmarks, detecting faults with ECCs, and testing with BIST.

The most standard flow is to run test patterns of BIST for both aging and testing, as shown in Figure 75(a). Such a basic test flow requires no modifications in the BIST and introduces no additional test patterns in the test flow. However, wearout resulting from continuously running test patterns for BIST does not reflect aging at real use conditions. Moreover, the choice of acceleration factors to project to use conditions is not straightforward, as the cause of failures is unknown.

An improved test flow [47] uses BIST only for testing, combined with commercial benchmarks or user-specific workloads for mimicking real usage scenarios, as illustrated in Figure 75(b). Testing with only BIST, however, fails to account for the tolerance of the system provided by ECCs which are widely used to enhance the reliability of a memory system. In addition, because the cause of failures is still unknown, the appropriate acceleration factors needed to estimate lifetime at use conditions are unknown.

We propose a new test flow that uses ECCs to detect failures while the circuit/system is running benchmarks during ALT, shown in Figure 75(c). Because of the use of ECCs to detect faults, new problems arise. ECCs failures could be either hard or soft faults. In order to distinguish wearout failures from soft failures, we deploy a sequence of commands proposed in recent work [101].

It is necessary to separate the sequence of faults from each wearout mechanism so that proper acceleration factors can be determined. To separate the sequence of faults from each wearout mechanism, it is necessary to diagnose the cause of each failure. Lifetime estimation for the full system is then estimated separately for each TDDB mechanism by computing the appropriate acceleration factors, that is, Weibull slope and characteristic lifetime. The distributions are then scaled to use conditions with the acceleration factors and combined to determine lifetime distribution of the circuit/system at the use condition, while accounting for all wearout mechanisms. Obviously, the most significant cost of this approach is the physical failure analysis required to identify the cause of wearout faults as needed to determine the appropriate acceleration factors.

One key advantage of the use of reliability simulation is that it can be used to select test conditions for the circuit/system-level test of each wearout mechanism. If test conditions are selected that are especially sensitive to each wearout mechanism, physical failure analysis can be avoided, saving significant time and cost.

5.2.4 Defining the Test Domain of System-Level ALT for Time-Dependent Dielectric Breakdown

We investigate how we design of experiments for various system-level accelerated life tests using simulation. Before we design experiments, we need to examine failure statistics of the memory system at accelerated stress conditions, which provides insight into optimizing the design of experiments (DOEs) for system-level ALT. Assuming a 14-day test, we simulate the probability of failure caused by all TDDB mechanisms. From failure trends at various stress conditions, we observe several factors that we should take into account to define test domains for the system-level accelerated life test: detectability, lifetime variation coverage, and selectivity.

5.2.4.1 Detectability of System-Level ALT

5.2.4.1.1 The lower bounds on P_{fail} to ensure detectability

Failures at stress conditions should be detected by BIST or ECCs within a limited test time at the specified stress conditions. To extract failure statistics from experiments, we must detect at least one failure at the test conditions within the test time. The minimum detectable number of failures defines the lower bound on the acceptability region in terms of detectability, shown in Figure 76. Since system-level ALT aims to obtain the reliability characteristics of a memory system, the lower bound on P_{fail} for each wearout mechanism is determined by the capacity of a memory system. Since we investigate a memory system in an open-source microprocessor of Leon3, containing 226 *K* 6T SRAM cells, the minimum failure rate is 4.3×10^{-6}.

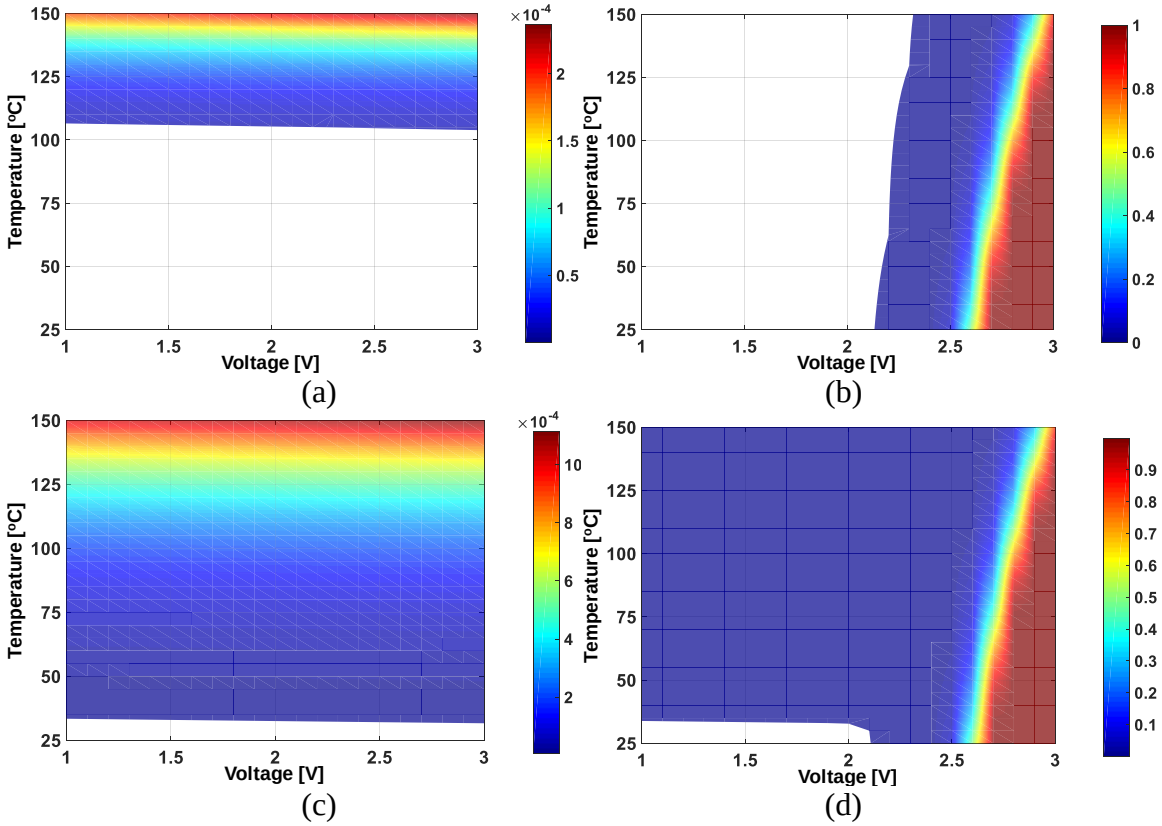


Figure 76 Defining the acceptability region of each time-dependent dielectric breakdown mechanism taking detectability into account by limiting the minimum probability of failure: (a) BTDDB, (b) GTDDB, (c) MTDDB, and (d) TDDB. The scale indicates the probability of failure of the SRAM. White areas indicate test conditions where the target mechanism is unlikely to cause any cells in the SRAM to fail.

5.2.4.1.2 The upper bounds of P_{fail} in terms of detectability

While the sensitivity of a BIST scheme or ECCs determines the lower bound of detectability, the capability of ECCs determines the upper bound of P_{fail}, depicted in Figure 77. If the number of failures in a memory is beyond the capability of ECCs, the system is not operational, and system-level ALT cannot be conducted. We investigate ECCs with both single-bit error correction, commonly used in a memory system, and double-bit error correction, an example of stronger ECCs than the conventional SEC.

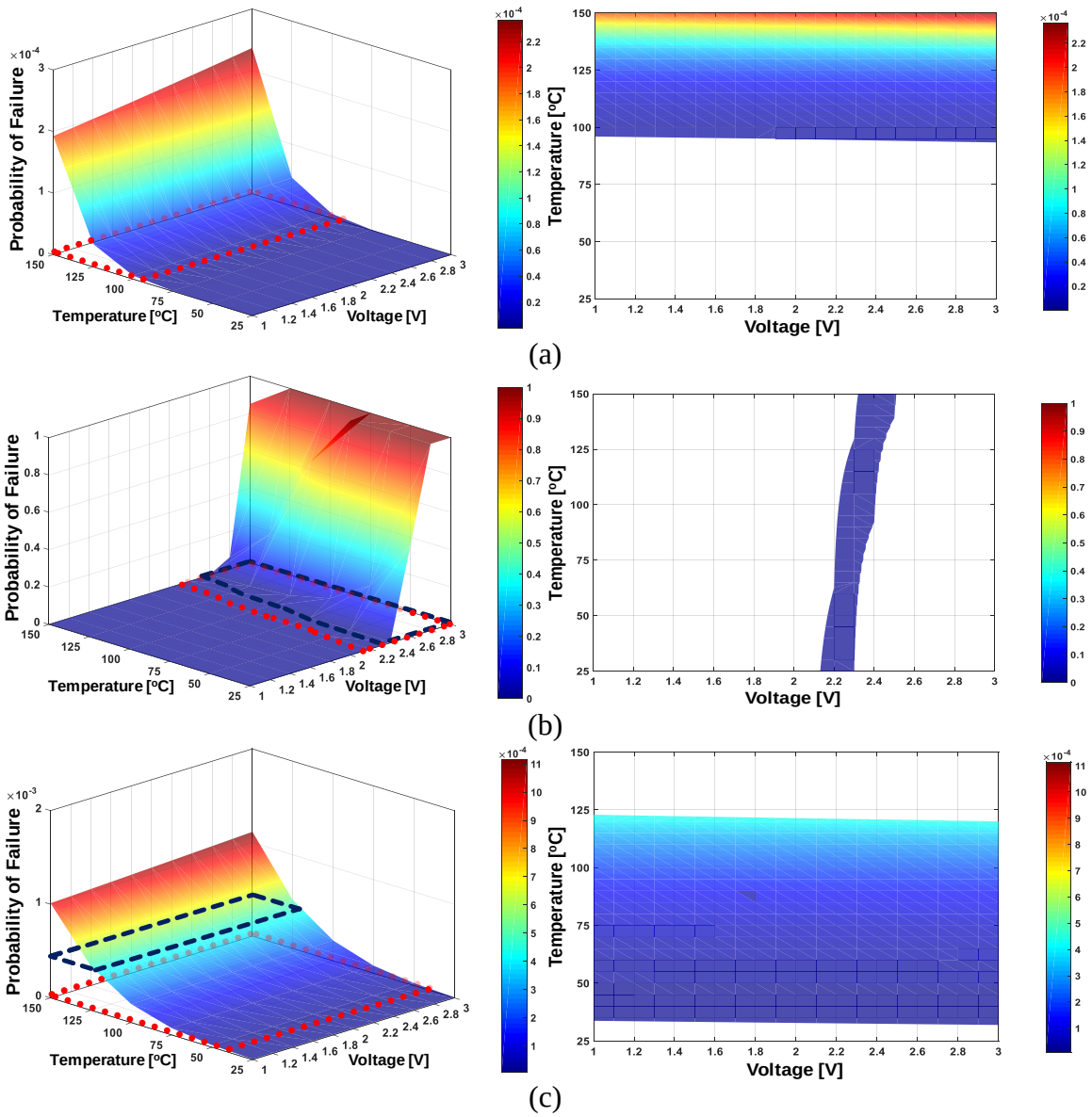


Figure 77 Simulated probabilities of failure with various temperatures and voltages with a 14-day test to define the acceptability regions for each TDDB mechanism by taking detectability into account, limited by the minimum and the maximum probability of failure: (a) BTDDB, (b) GTDDB, and (c) MTDDB. Red dotted lines show the lower bounds on the probability of failure, while blue dotted lines depict the upper bounds on the probability of failure.

As for the memory configuration, we assume that the size of a word is 32 bits, and the number of words (N_w) is 7,232. Since a word fails if two or more errors under SEC occur in one word, we calculate the probability of a word failure, $P_{w,SEC}$, using the binomial distribution as

$$P_{w,SEC} = 1 - (1 - P_b)^{n_b} - n_b P_b (1 - P_b)^{n_b - 1}, \quad (49)$$

where P_b and n_b denote the probability of a bit failure in a memory system and the number of bits in a word, respectively. Therefore, if a memory system has more than 123 errors, we expect that a double-bit error in a word can happen, which is beyond the capability of SEC ECCs.

We also examine a case of DEC ECCs. With DEC, a word fails if three or more errors occur in a word. We calculate the probability of a word failure, $P_{w,DEC}$, using the binomial distribution as

$$P_{w,DEC} = P_{w,SEC} - C_2^{n_b} \cdot {P_{bit}}^2 (1 - P_b)^{n_b - 2}. \quad (50)$$

Therefore, if a memory system has more than 718 errors, we expect that a triple-bit error in a word will occur, which is beyond the capability of DEC ECCs. The upper bounds defined by P_{fail} correspond to the capability of DEC ECCs

5.2.4.1.3 The acceptability regions considering detectability

The acceptability regions for each wearout mechanism corresponding to both the lower and the upper bounds on P_{fail} in terms of detectability are plotted on the two-dimensional test domains for both SEC and DEC ECCs in Figure 77. Red dotted lines denote the lower bound on P_{fail}, that is, the minimum failure rate, and blue dotted lines represent the upper bound on P_{fail}, that is, the maximum failure rate.

Figure 78 depicts the acceptability regions for TDDB for system-level ALT with the BIST scheme without ECCs, SEC, and DEC. It can be seen that DEC increases the upper bounds and enlarges the acceptability regions. However, note that the lower bounds for P_{fail} for BIST, SEC, and DEC are identical by comparing Figure 76(d) and Figure 78.

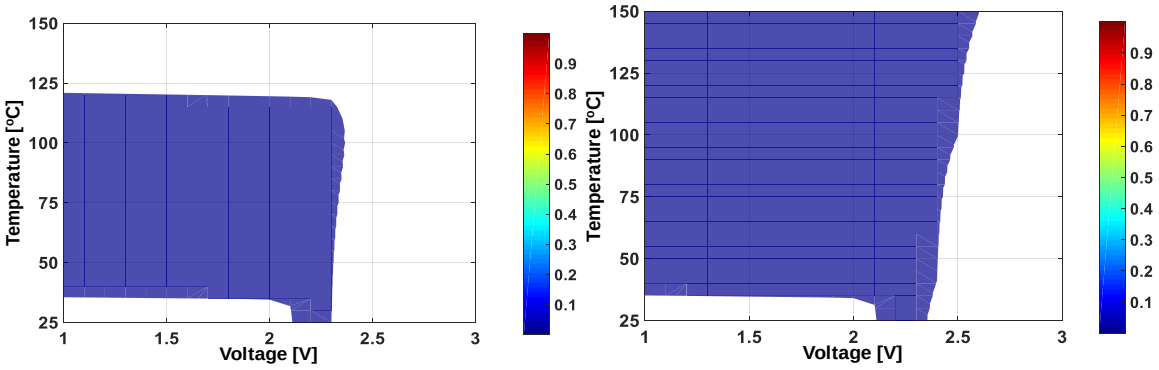


Figure 78 Defined acceptability regions for TDDB with a 14-day test by taking detectability into account after applying both the lower and the upper bounds to the probability of failure resulting from TDDB, combining all three mechanisms: (a) BIST-only without ECCs, (b) BIST with SEC, and (c) BIST with DEC.

5.2.4.2 Lifetime Variation Coverage of the Accelerated Life Test

In the reliability simulations, we assume a maximum range of ±30 % process variation. Table 5 summarizes the ranges of process variation parameters that we have employed. As shown in Figure 79, process variation results in variation in the probability of failure, which eventually causes variation in system lifetime. Figure 80 depicts variation in the probability of failure as a function of the process variation range from ±5 % to ±30 %, normalized to the variation in the probability of failure with ±30 % process variation. In testing, we may use various test samples with various process corners. Process variation causes variation in the probability of failure, which eventually variates the lifetime of a system. To ensure all variations in the lifetime of a system caused by process variation using testing, we investigate lifetime variation coverage of a test condition for each TDDB

Table 5 Process Variation Parameters

Variation	Range	Variation	Range
Gate length (ΔL_{GTDDB})	±30 %	Gate-Contact space(ΔS_{MTDDB})	±30 %
Gate width (ΔW_{GTDDB})	±30 %	Vulnerable length of dielectric segment($\Delta L_{M/BTDDB}$)	±30 %
Back-end dielectric space (ΔS_{BTDDB})	±30 %	Space between metal lines (ΔS_{BTDDB})	±30 %
Gate length (ΔL_{GTDDB})	±30 %	Gate-Contact space(ΔS_{MTDDB})	±30 %

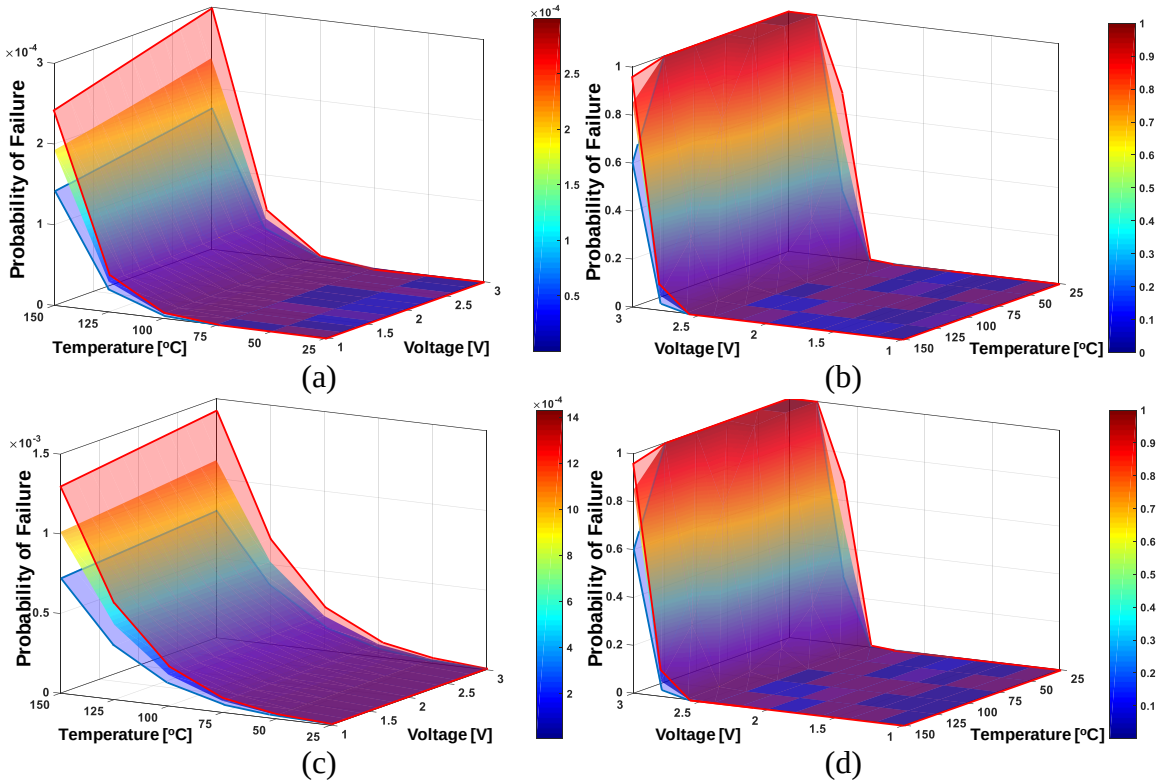


Figure 79 Simulated probability of failure with ±30 % process variation resulting from: (a) BTDDB, (b) GTDDB, (c) MTDDB, and (d) all TDDB mechanisms combined. The red line represents the probability of failure with +30 % process variation while the blue line plots the probability of failure with -30 % process variation.

mechanism. We define lifetime variation coverage of a test condition as normalized lifetime variation of a wearout mechanism at a test condition to maximum lifetime variation over all test domains with ±30% process variation.

Lifetime variation coverage varies with process variation and the probability of failure of a wearout mechanism. As process variation declines, lifetime variation coverage also decreases. At a low probability of failure, such lifetime variation coverages of GTDDB, BTDDB, and MTDDB are identical at the same process variation range, as shown in Figure 80. However, at a higher failure probability, we observe that the lifetime variation coverage of a TDDB mechanism at a given process variation range decreases. As the stress increases, that is, at higher temperatures and voltages, the probability of failure increases exponentially so that the contribution of process variation to variation in the probability of failure becomes relatively small. This reduced sensitivity of the probability of failure to process variation lowers the lifetime variation coverage.

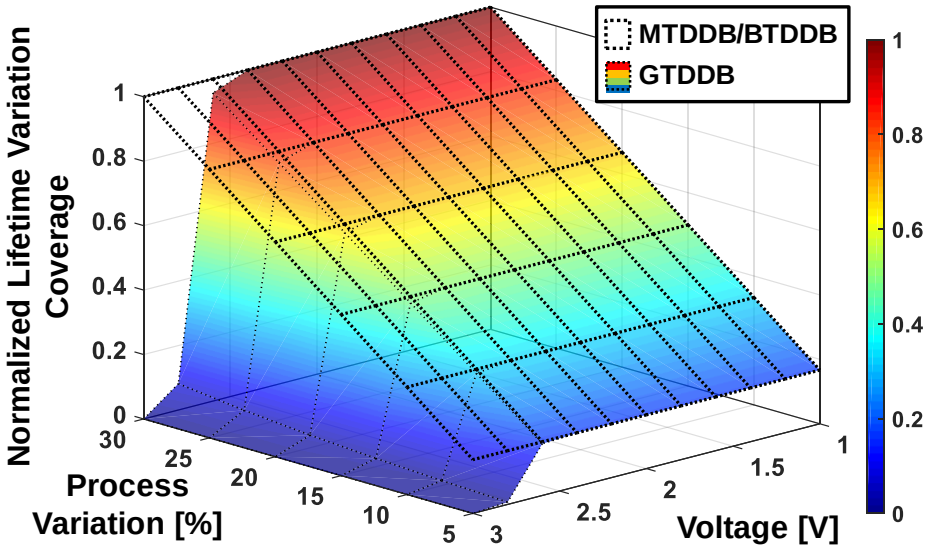


Figure 80 Variation in the probability of failure resulting from each TDDB wearout mechanism with various levels of process variation normalized to a maximum variation of the probability of failure for each wearout mechanism caused by ±30 % process variation with a 14-day test at 125 ºC.

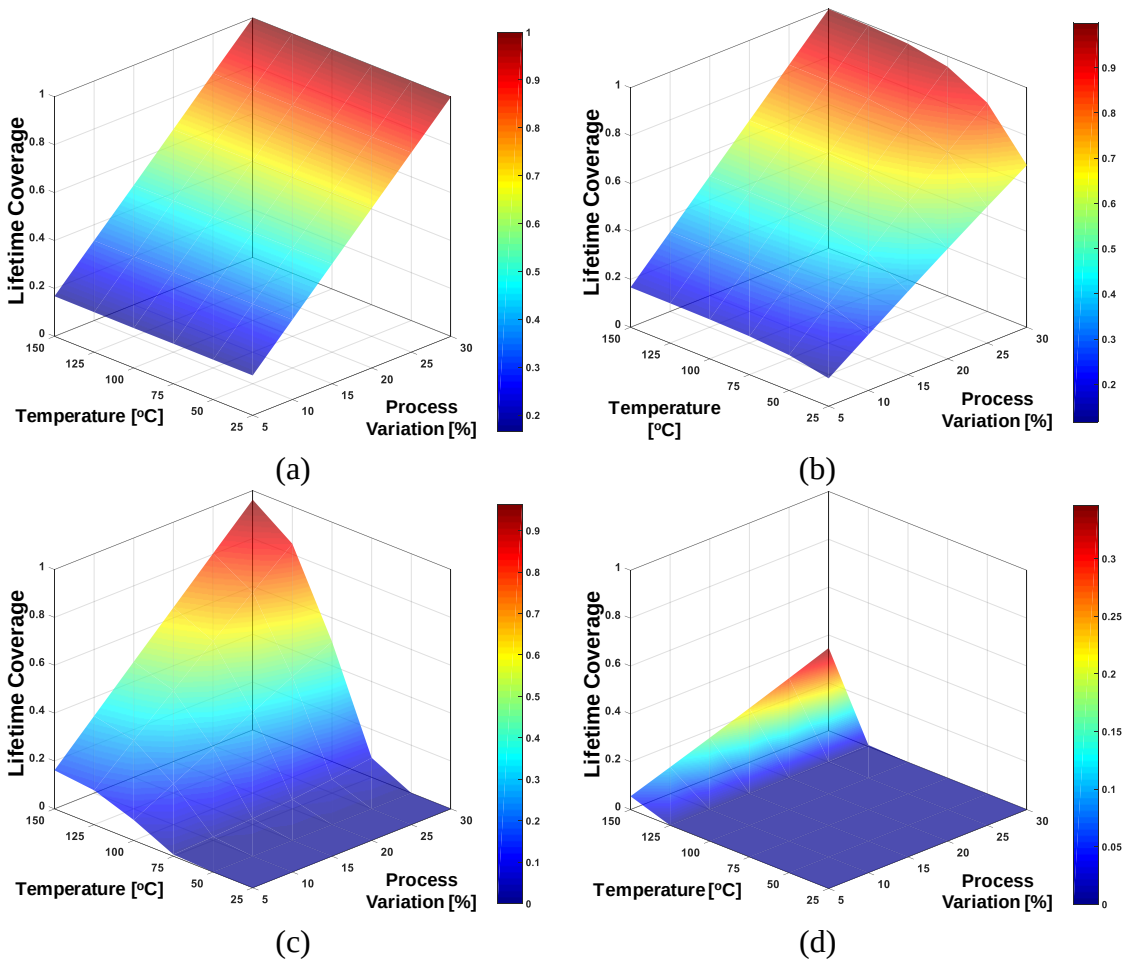


Figure 81 Simulated lifetime variation coverage degraded by GTDDB with various levels of process variation at various temperatures from 25 ºC to 150 ºC and voltages: (a) from 1 V to 2.4 V, (b) 2.6 V, (c) 2.8 V, and (d) 3 V.

Figure 81 shows that lifetime variation coverage of GTDDB varies as a function of voltage acceleration. At a low probability of failure from 1 V to 2.5 V, the slope of the lifetime variation coverage with respect to the process variation range remains constant. However, at a higher voltage than 2.5 V, which results in a higher probability of failure, testing cannot reflect the full range of process variation, which lowers the accuracy of lifetime estimates in the presence of process variation. Therefore, we should take lifetime variation coverage into account when optimizing the experimental design of system-level ALT. By eliminating the test domain that cannot yield 100 % process variation coverage, we re-define acceptability regions for each TDDB mechanism in Figure 82.

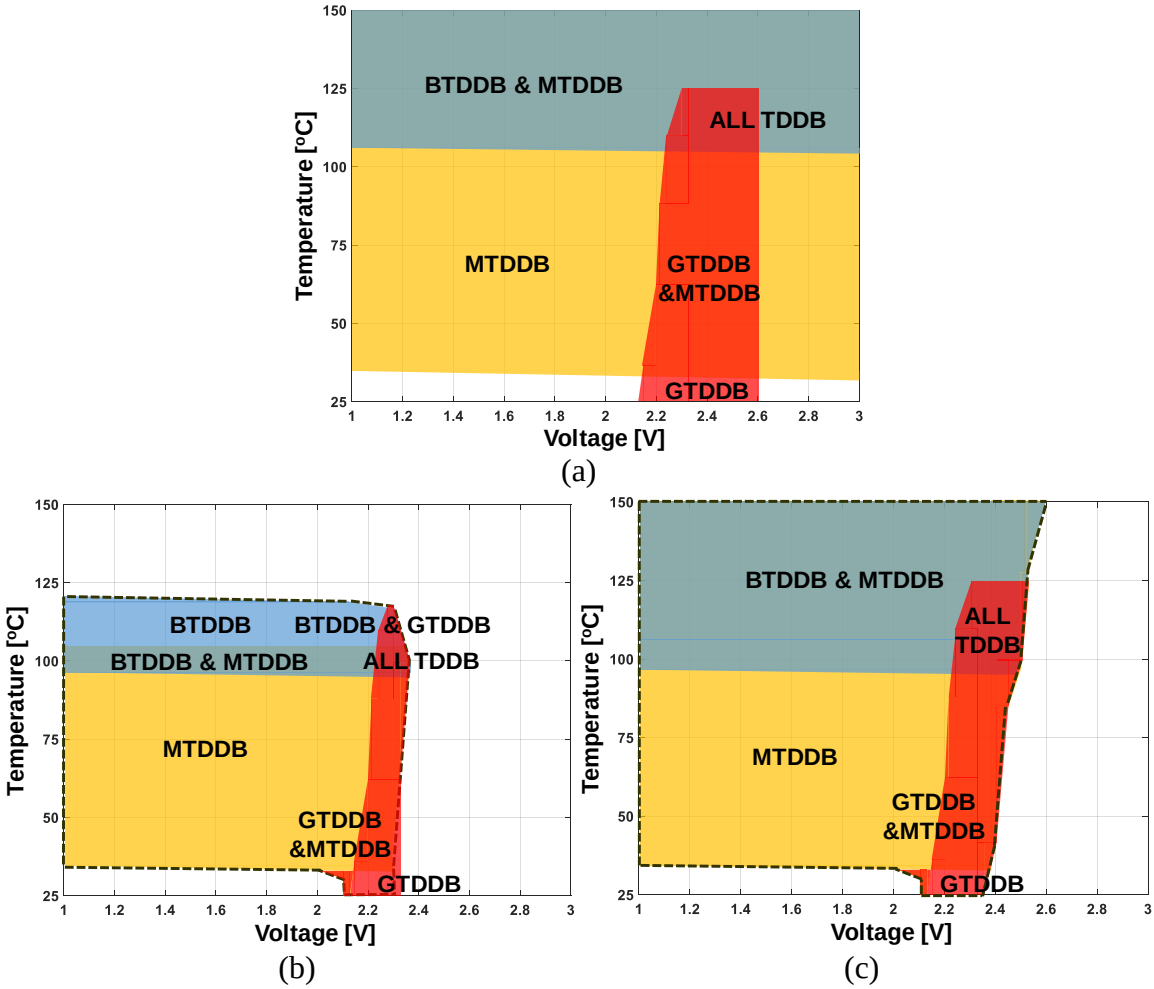


Figure 82 (a) Acceptability regions for TDDB after a 14-day test, taking the coverage of lifetime variation into account by eliminating test conditions that cannot reflect the total amount of lifetime variation caused by process variations: (a) BIST-only without ECCs, (b) BIST with SEC, and (c) BIST with DEC.

5.2.4.3 Selectivity of TDDB Mechanisms

In an acceptability region for each TDDB mechanism, for better distinguishing the failures degraded by each TDDB mechanism from failures caused by other TDDB mechanisms, failures associated with a target TDDB mechanism must outnumber failures from other wearout mechanisms. To quantify the accuracy of a test condition in isolating a target failure mechanism from others, we define the ratio of the number of failures caused by the target wearout mechanism to the total number of failures at a specific test condition

as selectivity. If selectivity is high, we need no physical failure analysis that diagnoses the cause of failure, which in turn results in substantial savings in time and cost when using the resulting data to estimate circuit/system lifetime.

Figure 83 depicts optimal test regions of the TDDB mechanisms in terms of selectivity as a function of stress conditions. In an acceptability region with a selectivity of '1,' which exists for GTDDB and MTDDB, we expect to completely isolate failures caused by the target mechanism from failures resulting from others. However, note that the selectivity of BTDDB fails to reach '1.' This is because the probability of BTDDB failures does not dominate for any of the test conditions. Therefore, isolating failures due to BTDDB requires physical failure analysis.

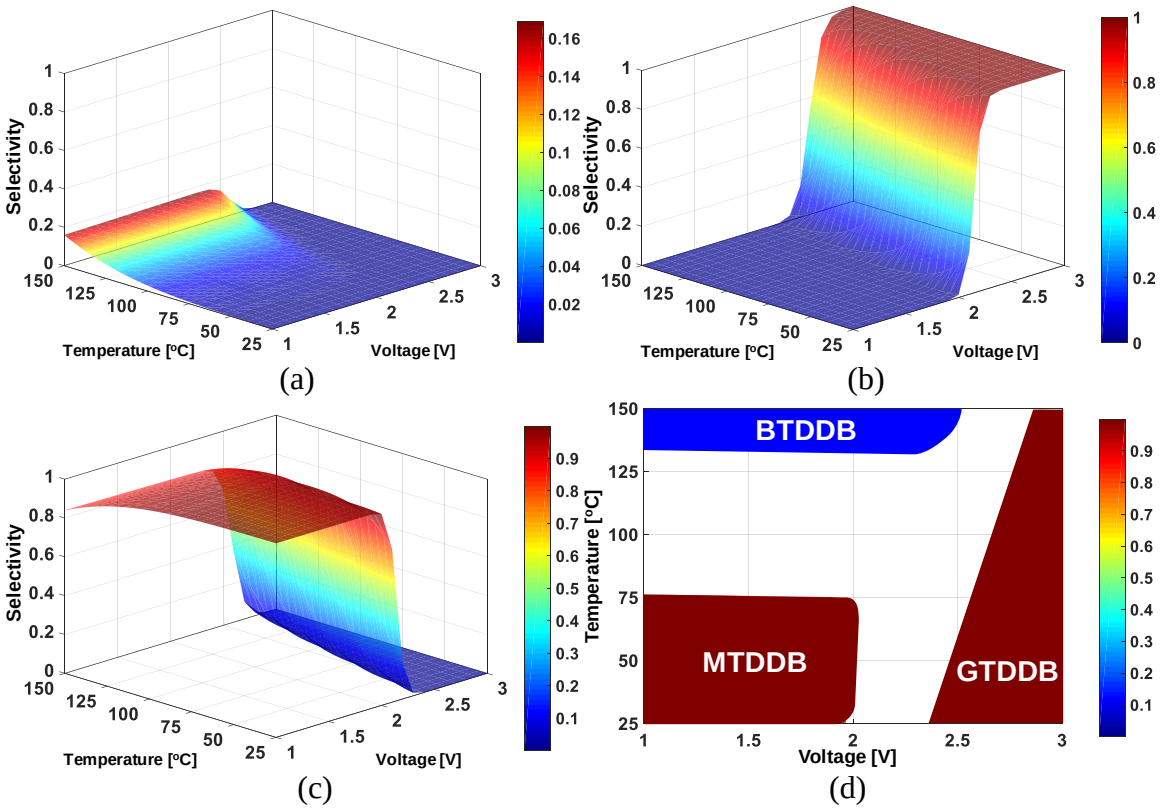


Figure 83 Optimal test regions of each TDDB wearout mechanism in terms of selectivity with a 14-day test: (a) BTDDB, (b) GTDDB, (c) MTDDB, and (d) all TDDB mechanisms.

5.2.5 Experimental Designs for System-Level ALT

After defining acceptability regions for each mechanism by accounting for detectability, lifetime variation coverage, and selectivity of TDDB mechanisms, we discuss how we design system-level ALT within the defined acceptability regions. Before discussing DOEs for ALT at the system level, we need to take a look at the conventional design of experiments for device-level accelerated life tests.

5.2.5.1 The Conventional Device-Level ALT Plan

The E-model has been widely used to model TDDB [1]-[8],[12],[13]:

$$t_{bd} = A \cdot e^{-\gamma E^m} \cdot e^{\frac{E_A}{kT}}, \tag{51}$$

where t_{bd} is the time to breakdown of a device, A and m are fitting parameters, γ is the electric field acceleration factor, k is Boltzmann's constant, E_A is the activation energy, T is the temperature, and E is the electrical field, which corresponds to V/s, where V and s denote the voltage and the space between two nodes adjacent to a dielectric, respectively. To determine the acceleration parameters, i.e. the electrical field acceleration factor and activation energy, γ and E_A, we conventionally choose several test points, which consist of two or more electrical field points at a fixed temperature and two or more temperature points at a fixed electrical field [12], as depicted in Figure 84.

5.2.5.2 System-Level ALT with DOEs

In system-level ALT, since we still need to extract field and activation energy acceleration factors, we exploit the conventional device-level DOEs as the system-level

DOEs for each wearout mechanism. Figure 85 illustrates the proposed device-level DOEs for TDDB mechanisms in their acceptability regions for all cases of system-level ALT, i.e., with BIST without ECCs, BIST with SEC, and BIST with DEC. The acceptability regions for MTDDB are identical for all schemes. For DEC ECCs, the acceptability region for BTDDB moves the test domain in the direction of higher selectivity for BTDDB, while the acceptability region for GTDDB for DEC becomes larger than that for SEC, which improves the accuracy of the system-level accelerated life test. A BIST scheme without ECCs has a wider acceptability region for GTDDB than that for the schemes with ECCs.

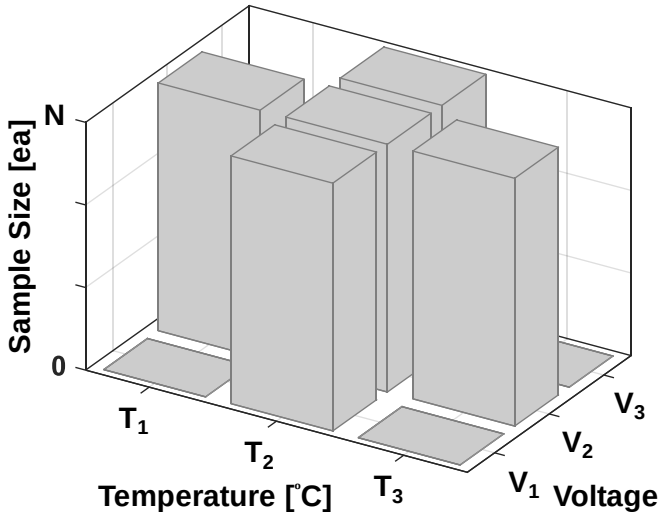


Figure 84 The conventional device-level accelerated lifetime test plan for estimating acceleration parameters for time-dependent dielectric breakdown models with the same number of samples at each test condition.

Using the method proposed in prior work [47], we optimize each DOE in the acceptability region for each TDDB mechanism so that the reliability of a DOE is maximized with a limited number of samples at each test point. Figure 86 shows simulated reliabilities of DOEs with various total sample sizes. In this simulation, we assume the same sample size at each of the test points in each acceptability region for each TDDB mechanism. Testing with BIST without ECCs shows the best results in terms of the reliability of the DOEs accounting for detectability, lifetime variation coverage, and

selectivity. The discrepancy between testing with a BIST scheme, with and without ECCs on the reliability of DOEs in Figure 86 represents the difference in the vulnerability of a memory system with and without ECCs to device-level wearout failures. ECCs with higher capability improve the accuracy of the DOEs, while requiring large area overheads and system modifications.

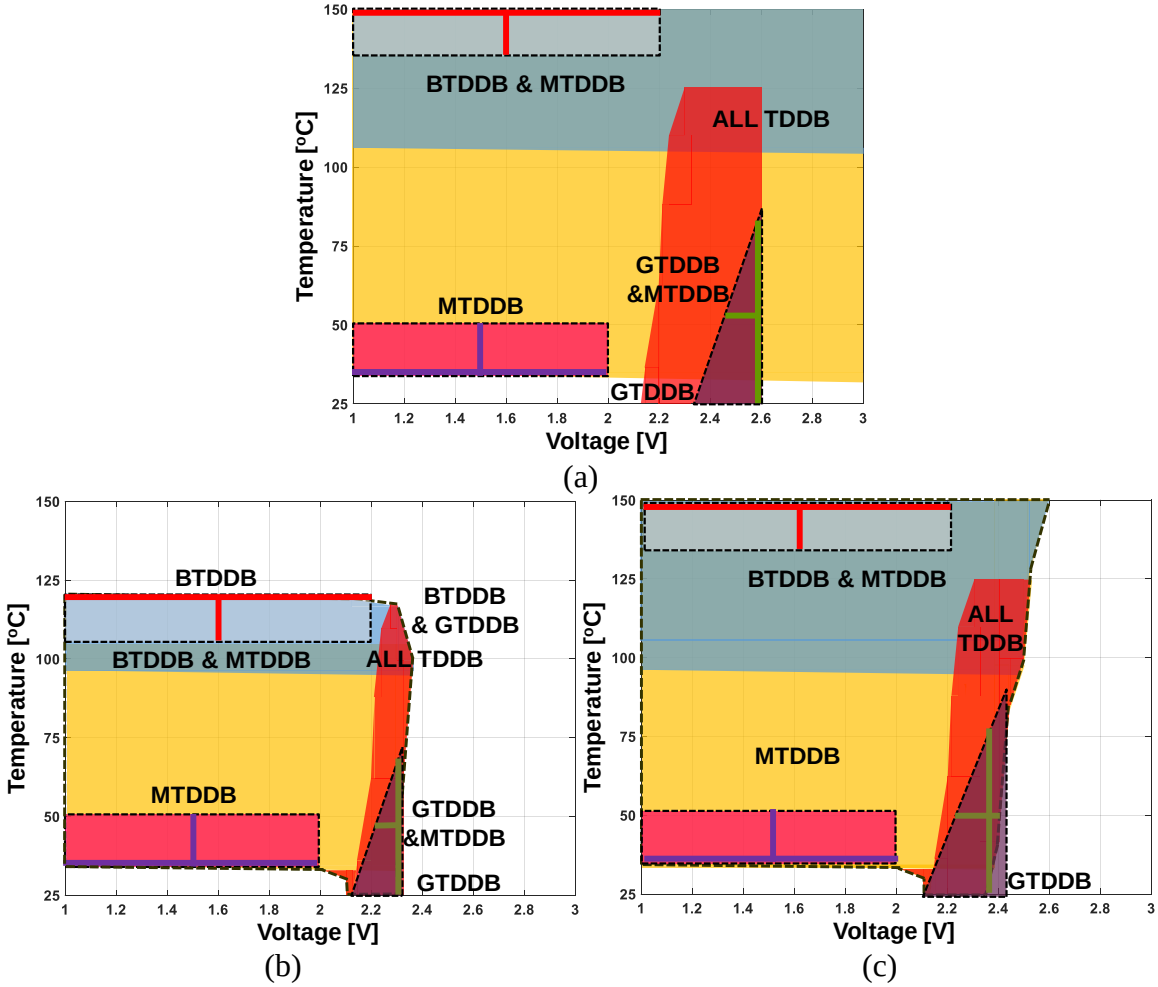


Figure 85 Proposed DOEs for each TDDB mechanism in each acceptability region for a system-level accelerated life test for (a) BIST without ECCs, (b) BIST with SEC, and (c) BIST with DEC with a 14-day test.

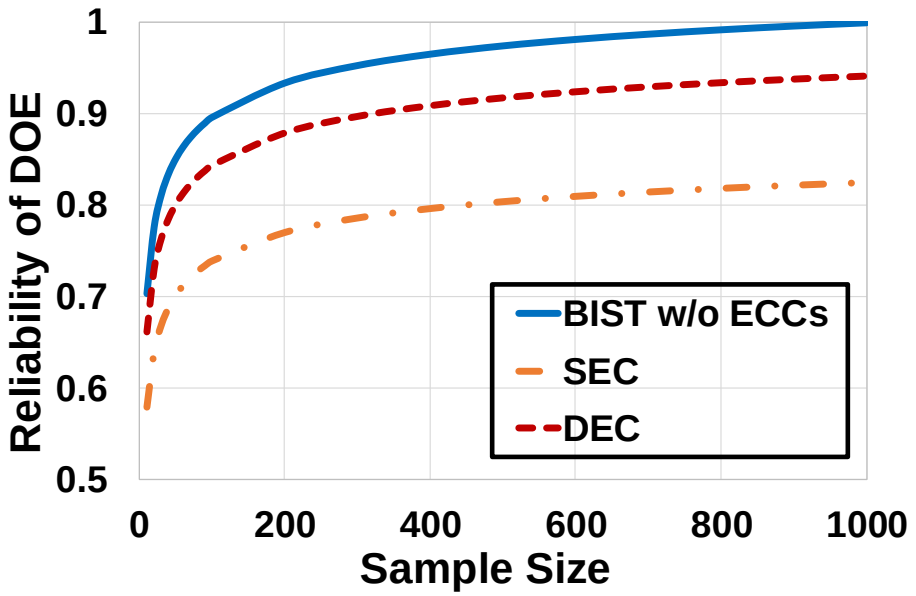
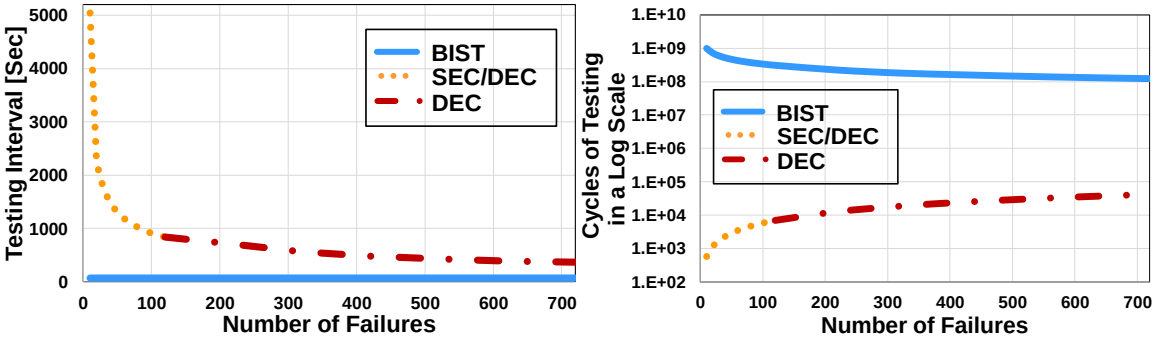


Figure 86 Simulated reliability of DOEs with various numbers of samples with the assumption that the actual process variation is ±30 %. We also assume that we place the same number of samples at each test condition for each test condition, i.e. BIST-only without ECCs, SEC, and DEC ECCs with a 14-day test.

5.2.6 Overheads of Testing

Figure 87 plots testing overheads of various system-level accelerated life tests without ECCs and with SEC and DEC ECCs. We assume that testing without ECCs periodically invokes testing every 68.36 seconds, producing ten wearout failures on average when the Weibull characteristic lifetime of the TDDB wearout distribution at the accelerated stress condition is 10^6 sec. Once testing without ECCs starts, the worst-case number of test cycles, proportional to (total memory cells – the number of wearout failures) assuming that the complexity of BIST test patterns is *O(N)*, should be executed because we assume the use of memoryless BIST. Therefore, the test interval is a constant as 68.36 seconds, and the number of test cycles gradually decreases as the number of wearout failures increases, as shown in Figure 87.



(a) (b)

Figure 87 Testing overheads of the BIST-only without ECCs, SEC, and DEC ECCs schemes with a 14-day test: (a) test interval and (b) test cycles.

Testing exploiting ECCs has decreasing test intervals, because the method is aware of the increasing number of wearout failures, as illustrated in Figure 87. Such a method also requires much fewer test cycles, which saves substantial test time at the beginning of the aging test. The number of test cycles is proportional to the number of wearout failures with the conservative assumption that all failures detected by ECCs are confirmed as wearout failures (rather than soft failures) by BIST.

5.2.7 Summary

For estimating system lifetime using realistic test patterns at accelerated conditions, we have proposed system-level accelerated life tests that exploit BIST and ECCs. Based on observations from system reliability simulation at various stress conditions, we define acceptability regions for testing of each TDDB mechanism.

In such acceptability regions, we employ and optimize the conventional device-level DOEs for each TDDB wearout mechanism separately to be able to extract each TDDB failure distribution using system-level ALT. We also extract acceleration factors for each TDDB wearout mechanism, and we ensure that all wearout mechanisms are visible under system-level ALT. Because test conditions are designed for each wearout mechanism,

diagnosis of the cause of failure is not always required, saving the time and cost associated with physical failure analysis. This work proposes a method to enhance the accuracy of estimating the lifetime of a memory system using system-level accelerated life test, while taking all TDDB wearout mechanisms into account.

CHAPTER 6. CONCLUSIONS AND FURTURE WORK

To achieve low cost per bit with low power and high performance, memory technology has continued to scale down to small feature sizes. However, cost-effectively developing an advanced fabrication technology that further shrinks feature sizes of transistor below 14 nm is challenging. Furthermore, such extremely reduced feature sizes of devices not only exacerbate several failure modes and but also create new failure modes in a memory system such as failures resulting from variable retention time, row hammering, and wearout.

The objective of the thesis is to develop design methodologies for scalable and reliable memory systems in the presence of such scalability and reliability issues. In this research, after investigating the origins and device-level models of memory failures, to examine the impact of such failures on operations of a memory system, this work proposes circuit- and system-level modeling and simulating methods. After investigating simulation results generated using the proposed modeling and simulating methods, this research proposes design methods that mitigate row-hammering phenomenon using counter-based or probabilistic row activations and repair increasing wearout failures by exploiting error-correcting codes for error detection and sequence of commands for error identification during field operations. To enhance the reliability of a memory system, this dissertation proposes methodologies that accurately estimate memory reliability degradation and diagnose system-level memory failures by employing system-level accelerated life test using built-in self-test and error correcting codes. This work also proposes a method that optimizes design of experiments for isolating a failure caused by a target wearout

mechanism from failures caused by the others and minimizing errors in the estimation of wearout parameters at the normal operating condition.

In chapter 2, this dissertation discusses the origins and device-level models of memory failures caused by variable retention time, row hammering, and wearout. This work claims that we should model variable retention time in DRAM by investigating random telegraph noise in not operating current but leakage current by accounting for random fluctuations in not only gate-induced drain leakage but also gate leakage. This research investigates row hammering phenomenon in DRAM by analyzing the row hammering threshold based on leakage current resulting from coupling between neighboring wordlines. This dissertation also introduces device-level wearout models used in system-level modeling and simulation.

In chapter 3, this research proposes circuit- and system-level modeling and simulation methodologies for investigating the impact of memory failures resulting from variable retention time, row hammering, and wearout on memory operations. Using the proposed circuit-level model by exploiting time-varying resistors for mimicking random fluctuations in leakage current, the work proposes the algorithm for simulating variable retention time in DRAMs. In addition, by simulating memory activation patterns, this work demonstrates that row hammering can occur during normal memory operations. Lastly, this research simulates the impact of wearout failures on memory operations. Using SPICE simulation, we investigate malfunctions caused by BTI, HCI, and GTDDB in eDRAMs, which motivated the development of a built-in self-test scheme of eDRAMs. We introduce circuit-level resistive models for TDDB in SRAMs and a system-level reliability estimation

methodology that employs a bottom-up approach to estimate the reliability degradation of a memory system from device-level degradation of each potential wearout defect location.

In chapter 4, this work proposes circuit and system design methodologies for scalable and reliable memory systems, mitigating memory failures due to row hammering and wearout in memory. Since an activation on a victim row within a refresh cycle restores original data integrity of DRAM cells, this work proposes counter-based and probabilistic row activation methods for tolerating failures resulting from row hammering. For extending lifetime and reliability of a memory system, with negligible area and performance overheads, this research proposes a BISR-like post-package repair scheme that exploits ECCs for fault detection and the proposed sequence of commands, that is, *Read-Write-Read-Compare* for correctable errors and *Read-Invert-Write-Read-Compare* for uncorrectable errors, for fault identification. By exploiting the conventional repair scheme implemented in DRAMs with small modifications, such detected and identified hard faults are repaired, extending the lifetime of a memory system.

In chapter 5, this dissertation proposes design methodologies that test and diagnose system-level wearout failures in memory. This research proposes a BIST scheme for detecting and diagnosing failures caused by BTI, HCI, and GTDDB in eDRAMs by employing a current detector and the proposed test patterns. To accurately estimate and characterize wearout failures at a system level, this work proposes system-level accelerated life test with BIST and error-correcting codes. System-level ALT with BIST that isolates a target mechanism from various TDDB mechanisms using optimized DOEs allows us to better characterize wearout distributions by accounting for circuit and system tolerance to device-level wearout failures. Furthermore, system-level ALT with ECCs helps us to

accurately estimate the lifetime of a memory system degraded by wearout mechanisms by aging memory with realistic usage scenarios and testing memory while accounting for tolerance of ECCs to wearout failures.

This dissertation proposes modeling, simulating, mitigating, and characterizing schemes of reliability issues in a memory system such as variable retention time, row hammering, and wearout. Verification and calibration of the proposed simulation methodologies based on silicon testing results would be essential to the improvement of the proposed research. Furthermore, even though this dissertation deals with BTI, HCI, GTDDB, MTDDB, and BTDDB, applying the proposed design methodologies to mitigating and characterizing failures resulting from all possible wearout mechanisms including electro-migration and stress migration would advance the proposed work. We can improve the accuracy of lifetime estimation using system-level accelerated life tests by developing test patterns that enhance the selectivity of a wearout mechanism at test stress conditions within a limited test time. Design of experiments of system-level ALT using various test groups with differences in benchmarks or test patterns for aging and numbers of test samples would also enhance the accuracy of estimating wearout characteristics of a memory system. We can optimize design of experiments for system-level accelerated life test by employing sampling strategy that is aware of process variation that would reduce sample size at each test point in acceptability regions by improving the accuracy of estimation. Since we may obtain testing results with various test time within a limited test time, temporal optimization of design of experiments for system-level accelerated life test would also improve the accuracy of testing.

REFERENCES

[1] C.-C. Chen, T. Liu, S. Cha, and L. Milor, "System-level modeling of microprocessor reliability degradation due to BTI and HCI," in Proc. Int. Reliability Physics Symp., 2014.

[2] S. Drapatz, G. Georgakos, and D. Schmitt-Landsiedel, "Impact of negative and positive bias temperature stress on 6T-SRAM cells," *Advances in Radio Science*, vol. 7, pp. 191-196, 2009.

[3] S. Bhardwaj, et al. "Predictive modeling of the NBTI effect for reliable design." in Proc. IEEE Custom Integrated Circuits Conference, 2006.

[4] C.-C. Chen, F. Ahmed, and L. Milor, "Impact of NBTI-PBTI on SRAMs within microprocessor systems: modeling, simulation, and analysis," *Microelectronics Reliability*, vol. 53, no. 9-11, pp. 1183-1188, Sept.-Nov. 2013.

[5] B. Kaczer, R. Degraeve, M. Rasras, K. Van de Mieroop, P. J. Roussel, and G. Groeseneken, "Impact of MOSFET gate oxide breakdown on digital circuit operation and reliability," *IEEE Transactions on Electron Devices*, vol. 49, no. 3, pp. 500-506, 2002.

[6] R. Rodriguez, J. H. Stathis, B. P. Linder, S. Kowalczyk, C. T. Chuang, R. V. Joshi, G. Northrop, K. Bernstein, A. J. Bhavnagarwala, and S. Lombardo, "The impact of gate-oxide breakdown on SRAM stability," *IEEE Electron Device Letters*, vol. 23, no. 9, pp. 559-561, 2002.

[7] R. Fernandez, J. Martin-Martinez, R. Rodriguez, M. Nafria, and X. Aymerich, "Analysis and modeling of a digital CMOS circuit operation and reliability after gate oxide breakdown: a case study," *Microelectronics Reliability*, vol. 42, no. 4-5, pp. 555-564, 2002.

[8] C.-C. Chen, S. Cha, and L. Milor, "System-level modeling of microprocessor reliablity degradation due to TDDB," in Proc. Design of Circuits and Integrated Systems, 2014.

[9] L. Milor, and C. Hong. "Backend dielectric breakdown dependence on linewidth and pattern density." *Microelectronics Reliability*, vol. 47, no. 9, pp. 1473-1477, 2007.

[10] C.-W. Hsu, Y.-K. Fang, W.-K. Yeh, C.-Y. Chen, Y.-T. Chiang, F.-R. Juang, C.-T. Lin, and C.-M. Lai, "Improvement of TDDB reliability, characteristics of HfO_2 high-k/metal gate MOSFET device with oxygen post deposition annealing," *Microelectronics Reliability*, vol. 50, no. 5, pp. 618-621, 2010.

[11] J.-G. Ahn, M. F. Lu, P.-C. Yeh, J. Chang, X. Wu, and S. Y. Pai, "Product-level reliability estimator with advanced CMOS technology," in Proc. IEEE Relia. Phys. Symp., 2013.

[12] F. Chen, C. Graas, M. Shinosky, K. Zhao, S. Narasimha, X. H. Liu, and C. Tian, "Breakdown data generation and in-die deconvolution methodology to address BEOL and MOL dielectric breakdown challenges," *Microelectronics Reliability*, vol. 12, part B, pp. 2727-2747, 2015.

[13] Z. Guan, M. Marek-Sadowska, and S. Nassif, "SRAM bit-line electromigration mechanism and its prevention scheme," in Proc. IEEE International Symposium Quality Electronic Design (ISQED), 2013.

[14] H. Tsuchiya, and Y. Shinji, "Electromigration lifetimes and void growth at low cumulative failure probability." *Microelectronics Reliability*, vol. 46, no. 9-11, pp. 1415-1420, 2006.

[15] C. Christiansen, Baozhen Li, J. Gill, R. Filippi, and M. Angyal "Via-depletion electromigration in copper interconnects." *IEEE Trans. Device and Materials Reliability*, vol. 6, no. 2, pp. 163-168, 2006.

[16] T. Oshima, K. Hinode, H. Yamaguchi, H. Aoki, K. Torii, T. Saito, K. Ishikawa, J. Noguchi, M. Fukui, T. Nakamura, S. Uno, K. Tsugane, J. Murata, K. Kikushima, H. Sekisaka, E. Murakami, K. Okuyama, and T. Iwasaki, "Suppression of stress-induced voiding in copper interconnects," in Proc. Int. Electron Devices Meeting, 2002.

[17] R. Wang, C.C. Lee, L.D Chen, K. Wu, and K.S. Chang-Liao, "A study of Cu/Low-k stress-induced voiding at via bottom and its microstructure effect," *Microelectronics Reliability*, vol. 46, no. 9, pp. 1673-1678, Oct. 2006.

[18] K. Yoshida, T. Fujimaki, K. Miyamoto, T. Honma, H. Kaneko, H. Nakazawa, and M. Morita, "Stress-induced voiding phenomena for an actual CMOS LSI interconnects," in Proc. IEEE International Electron Devices Meeting, 2002.

[19] F. Ahmed and L. Milor, "NBTI resistant SRAM design," in Proc. Int. Workshop on Advances in Sensors and Interfaces, 2011.

[20] F. Ahmed and L. Milor, "Reliable Cache Design with On-Chip Monitoring of NBTI Degradation in SRAM Cells using BIST," in Proc. IEEE VLSI Test Symp, 2010.

[21] F. Ahmed and L. Milor, "Analysis of on-chip monitoring of gate oxide breakdown in SRAM cells," *IEEE Trans. VLSI*, vol. 20, no. 5, pp. 855-864, May 2012.

[22] T. Kawagoe, J. Ohtani, M. Niiro, T. Ooishi, M. Hamada, and H. Hidaka, "A built-in self-repair analyzer (CRESTA) for embedded DRAMs," in Proc. IEEE Int. Test Conf., 2000.

[23] T.-H. Wu, P.-Y. Chen, M. Lee, B.-Y. Lin, C.-W. Wu, C.-H. Tien, H.-C. Lin, H. Chen, C.-N. Peng, and M.-J. Wang, "A memory yield improvement scheme combining built-in self-repair and error correction codes," in Proc. International Test Conference (ITC), 2012.

[24] C.-L. Su, Y.-T. Yeh, and C.-W. Wu, "An integrated ECC and redundancy repair scheme for memory reliability enhancement," in Proc. the Defect and Fault Tolerance in VLSI and Nanotechnology Systems Symposium (DFT), 2005.

[25] A. González, F. Latorre, and G. Magklis, "Processor microarchitecture: An implementation perspective," *Synthesis Lectures on Computer Architecture* 5.1, Morgan and Claypool eBooks, 2010.

[26] B. Schroeder, E. Pinheiro, and W.-D. Weber, "DRAM errors in the wild: a large-scale field study," in Proc. ACM Special Interest Group on Performance Evaluation (SIGMETRICS), 2009.

[27] V. Sridharan and D. Liberty, "A study of DRAM failures in the field," in Proc. International Conference for High Performance Computing, Networking, Storage and Analysis (SC), 2012.

[28] A. Hwang, I. Stefanovici, and B. Schroeder, "Cosmic rays don't strike twice: understanding the nature of DRAM errors and the implications for system design," in Proc. ACM Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2012.

[29] K. Sudan, N. Chatterjee, D. Nellans, M. Awasthi, R. Balasubramonian, and A. Davis, "Micro-pages: Increasing DRAM efficiency with locality-aware data placement," in Proc. ACM Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2010.

[30] D.-H. Kim, P. Nair, and M. K. Qureshi, "Architectural support for mitigating row hammering in DRAM memories," *IEEE Computer Architecture Letters (CAL)*, vol. 14, no. 1, Jan.-June 2015.

[31] P. Nair, D.-H. Kim, and M. Qureshi, "ArchShield: Architectural framework for assisting DRAM scaling by tolerating high error rates," in Proc. International Symposium on Computer Architecture (ISCA), 2013.

[32] C. H. Stapper, J. A. Fifield, and H. L. Kalter, "High-reliability fault-tolerance 16-Mbit memory chips," *IEEE Trans. Relia.*, vol. 42, no. 4, pp. 596-603, Dec. 1993.

[33] J.-K. Wee, K.-S. Min, J.-T. Park, S.-P. Lee, Y.-H. Kim, T.-H. Yang, J.-D. Joo, and J.-Y. Chung, "A Post-Package Bit-Repair Scheme Using Static Latches with Bipolar-Voltage Programmable Antifuse Circuit for High-Density DRAMs," *IEEE Journal of Solid-State Circuits* (*JSSC*), vol. 37, no. 2, pp. 251-254, Feb. 2002.

[34] K. V. Aadithya, A. Demir, S. Venugopalan, and J. Roychowdhury, "SAMURAI: An accurate method for modelling and simulating non-stationary random telegraph noise in SRAMs," in Proc. Design Automation and Test in Europe (DATE), 2011.

[35] K. Aadithya, S. Venogopalan, A. Demir, and J. Roychowdhury, "MUSTARD: A coupled, stochastic /deterministic, discrete/continuous technique for predicting the impact of Random Telegraph Noise on SRAMs and DRAMs," in Proc. Design Automation Conference (DAC), 2011.

[36] S. Irobi, Z. Al-Ars, and S. Hamdioui, "Detecting memory faults in the presence of bit line coupling in SRAM devices," in Proc. IEEE International Test Conference, 2010.

[37] H. J. Oh, D. S. Woo, Y. S. Lee, D. H. Kim, S. E. Kim, G. W. Ha, H. J. Kim, N. J. Kang, J. M. Park, Y. S. Hwang, D. I. Kim, B. J. Park, M. Huh, B. H. Lee, S. B. Kim, M. H. Cho, M. Y. Jung, Y. J. Kim, C. Jin, D. W. Shin, M. S. Shim, C. S. Lee, W. S. Lee, J. C. Park, G. Y. Jin, Y. J. Park, and K. Kim, "S-RCAT (sphere-shaped-recess-channel-array transistor) technology for 70nm DRAM feature size and beyond," in Proc. VLSI Symp. Dig. Tech. Papers, 2005, pp. 34–35.

[38] M. S. Yoo, K. S. Choi, and W. K. Sun, “Saddle-fin cell transistors with oxide etch rate control by using tilted ion implantation (TIS-Fin) for sub-50-nm DRAMs,” *J. Korean Phys. Soc.*, vol. 56, no. 2, pp. 643–647, 2010.

[39] S. Hong, “Memory technology trend and future challenges,” in Proc. IEEE Int. Electron Devices Meeting, 2010, pp. 12.4.1–12.4.4.

[40] S.Cha, C.-C. Chen, and L. Milor, "Frontend wearout modeling from device to system with power/ground signature analysis," in Proc. Int. Inte. Reli. Workshop, 2014.

[41] D. Schroder and J. Babcokc, "Negative bias temperature instability: Road to cross in deep submicron silicon semiconductor manufacturing," *J. Appl. Phys*. 2003; 94(1): 1-18.

[42] H. Wang, M. Miranda, F. Catthoor, and W. Dehaene, "Impact of random soft oxide breakdown on SRAM energy/delay drift," *IEEE Trans. Device Mater. Rel.* 2007; 7(4): 581-591.

[43] E. Wu, J. Stathis, B. Li, B. Linder, K. Zhao, and G. Bonilla, "A critical analysis of sampling-based reconstruction methodology for dielectric breakdown systems (BEOL/MOL/FEOL)," in Proc. IEEE Relia. Phys. Symp., 2015.

[44] J.-G. Ahn, M. F. Lu, N. Navale, D. Graves, P.-C. Yeh, J. Chang, and S. Y. Pai, "Product-level reliability estimator with budget-based reliability management in 20nm technology," in Proc. IEEE Relia. Phys. Symp., 2015.

[45] D.-H. Kim and L. Milor, "TDDB-Emerald: A Methodology for Estimating Memory Reliability Degradation Resulting from Time-Dependent Dielectric Breakdown," in Proc. IEEE Des. of Circuits and Integr. Syst., 2016.

[46] W. Song, S. Mukhopadhyay, and S. Yalamanchili, "Managing performance-reliability tradeoffs in multicore processors," in Proc. IEEE Relia. Phys. Symp., 2015.

[47] D.-H. Kim and L. Milor, "Memory Reliability Estimation Degraded by TDDB Using a Circuit-Level Accelerated Life Test," in Proc. IEEE Relia. Phys. Symp., 2017.

[48] Y. H. Lee, N. Mielke, M. Agostinelli, S. Gupta, R. Lu, and W. McMahon, "Prediction of Logic Product Failure Due to Thin-Gate Oxide Breakdown," in Proc. IEEE Relia. Phys. Symp., 2006.

[49] M. Chang, J. Lin, S. Shih, T.-C. Wu, B. Huang, J. Yang, and P.-I. Lee, "Impact of gate-induced drain leakage on retention time distribution of 256 Mbit DRAM with negative wordline bias," *IEEE Transactions on Electron Devices*, Vol. 50, No. 4, April 2003.

[50] H. Kim, K. Kim, T.-K. Oh, S.-Y. Cha, S.-J. Hong, S.-W. Park, and H. Shin, "RTS-like fluctuation in gate induced drain leakage current of saddle-fin type DRAM cell transistor," in Proc IEEE International Electron Devices Meeting (IEDM) 2009.

[51] B. Oh, H.-J. Cho, H. Kim, Y. Son, T. Kang, S. Park, S. Jang, J.-H. Lee, and H. Shin, "Characterization of an oxide trap leading to random telegraph noise in gate-induced drain leakage current of DRAM cell transistors," *IEEE Trans Electron Devices*, vol. 58, no. 6, pp. 1741-1747, June 2011.

[52] H.-J. Cho, S. Lee, B.-G. Park, and H. Shin, "Extraction of trap energy and location from random telegraph noise in gate leakage current (Ig RTN) of metal–oxide semiconductor field effect transistor (MOSFET)," *Solid-State Electronics,* vol. 54, Issue 4, pp. 362–367, April 2010.

[53] H.-J. Cho, Y. Son, B.-C. Oh, S. Lee, J.-H. Lee, B.-G. Park, and H. Shin, "Study on time constants of random tele-graph noise in gate leakage current through hot-carrier stress test," *IEEE Electron Device Letters*, vol. 31, no.9, pp. 1029-1031, 2010.

[54] M. Gurfinkel, J. S. Suehle, J. B. Bernstein, and Y. Shapiral, "Enhanced gate induced drain leakage current in HfO_2 MOSFETs due to remote interface trap-assisted tunneling," in Proc. International Electron Devices Meeting (IEDM), 2006.

[55] R. Yamada, and T. King, "Variable stress-induced leakage current and analysis of anomalous charge loss for flash memory application," in Proc. IEEE International Reliability Physics Symposium (IRPS), 2003.

[56] K. Roy, S. Mukhopadhyay, and H. Mahmoodi-Meimand, "Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits," *Proceedings of the IEEE*, vol. 91, no. 2, pp. 305-327, Feb. 2003.

[57] E. Rosenbaum and L. Register, "Mechanism of stress-induced leakage current in MOS capacitors," *IEEE Transactions on Electron Devices*, vol. 44, no. 2, pp. 317-323, 1997.

[58] M. Redeker, B. F. Cockburn, and D. G. Elliott, "An investigation into crosstalk noise in DRAM structure," in Proc. IEEE Int. Workshop Memory Tech.,Des. Testing, 2002, pp. 123–129.

[59] D.-S. Min and D.W. Langer, "Multiple twisted dataline techniques for multigigabit DRAM's," *IEEE J. Solid-State Circuits*, vol. 34, no. 6, pp. 856–865, 1999.

[60] A. Van Der Wel, E. Klumperink, J. Kolhatkar, E. Hoekstra, M. Snoeij, C. Salm, H. Wallinga, and B. Nauta Low-frequency noise phenomena in switched MOSFETs. *IEEE Journal of Solid-State Circuits*, vol. 42, no. 3, pp. 540-50, 2007.

[61] S. Machlup, "Noise in semiconductors: spectrum of a two-parameter random signal," *Journal of Applied Physics*, vol. 25, no. 3, pp. 341-343, 1954.

[62] P. Lai, J. Xu, W. Wong, H. Lo, and Y. Cheng, "Correlation between hot-carrier-induced interface states and GIDL current increase in n-MOSFET's," *IEEE Trans Electron Devices*, vol. 45, no. 2, pp. 521-8, 1998.

[63] R. Entner, A. Gehring, H. Kosina, T. Grasser, and S. Selberherr, "Impact of multi-trap assisted tunneling on gate leakage of CMOS memory devices," in Proc. NSTI Nanotech, 2005.

[64] T. Wang, T.-E. Chang and C. Huang, "Interface trap induced thermionic and field emission current in off-state MOSFET's," in Proc. IEEE International Electron Devices Meeting (IEDM), 1994.

[65] Y. Taur and T. H. Ning, *Fundamentals of Modern VLSI Devices*, 2nd ed. New York, NY, USA: Cambridge Univ. Press, 2009.

[66] X. Li, J. Qin, and J. Bernstein, "Compact Modeling of MOSFET Wearout Mechanisms for Circuit-Reliability Simulation," *IEEE Trans Dev and Mat Rel*, vol. 8, no. 1, pp. 98-121, 2008.

[67] G. Ribes, J. Mitard, M. Denais,S. Bruyere,F. Monsieur,C. Parthasarathy,E. Vincent, and G. Ghibaud, "Review on High-k Dielectrics Reliability Issues," *IEEE Trans on Device and Materials*, vol. 5, no. 1, pp. 5-18, 2005.

[68] J. W. Lee, B. H. Lee, H. Shin, and J. H. Lee, "Investigation of Random Telegraph Noise in Gate-Induced Drain Leakage and Gate Edge Direct Tunneling Currents of High-k MOSFETs," *IEEE Trans Electron Devices*, vol. 57, no. 4, pp. 913-8, 2010.

[69] R. Balasubramonian, M. Shevgoor, S. H. Pugsley, A. N. Udipi, A. Shafiee, K. Sudan, M. Awasthi, and Z. Chishti, "Usimm: The utah simulated memory module," 2012.

[70] 2012 Memory scheduling championship (MSC) [Online]. Available: http://www.cs.utah.edu/ rajeev/jwac12/, 2012.

[71] (2010). MT41J512M4:8Gb QuadDie DDR3 SDRAM Rev. A 03/11, Micron.

[72] J. Karidis, M. Franceschini, V. Srinivasan, L. Lastras, and B. Abali, "Enhancing lifetime and security of pcm-based main memory with startgap wear leveling," in Proc. IEEE/ACM Int. Symp. Microarchitect., pp. 14–23, 2009.

[73] Y. Kim, R. Daly, J. Kim, C. Fallin, J. H. Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu, "Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors," in Proc ACM/IEEE Int. Symp., Comput. Architect. (ISCA), 2014.

[74] J. Barth, D. Plass, E. Nelson, C. Hwang, G. Fredeman, M. Sperling, A. Mathews, T. Kirihata, W. Reohr, K. Nair, and N. Cao, "A 45nm SOI Embedded DRAM Macro for POWER^{TM} Processor 32MByte On-Chip L3 Cache," *J. Solid State Circuits*, vol. 46, no. 1, pp. 64-75, 2011.

[75] J. Safran, B. He, D. Leu, M. Yin, T. Weaver, A. Vehabovic, Y. Sun, A. Cestero, B. Himmel, G. Maier, C. Kothandaraman, D. Fainstein, J. Barth, N. Robson, T. Kirihata, K. Rim, and S. Iyer, "3D Stackable 32nm High-K/Metal gate SOI Embedded DRAM Prototype," in Proc. IEEE Symp. VLSI Circuits, 2011.

[76] Huard V., Parthasarathy C., Rallet N., Guerin C., Mammase M., Garge D., and Ouvard C, "New characterization and modeling approach for degradation from transistor to product level," in Proc. Int. Elec Dev Meeting (IEDM), 2007.

[77] S. Cha, C.-C. Chen, and L. Milor, "System-level estimation of threshold voltage degradation due to NBTI with I/O measurements," in Proc. Int. Reli Phys Symp (IRPS), 2014.

[78] R. Silva and G. Wirth, "Logarithmic behavior of the degradation dynamics of metal oxide semiconductor devices," *J. Stat. Mech., Theory Exp.*, vol. 4, pp. 4-25, 2010.

[79] E. Takeda and N. Suzuki, "An empirical model for device degradation due to hot-carrier injection," *IEEE Electron Device Letters*, vol. 4, no. 4, pp. 111-3, 1983.

[80] J. Stathis, "Percolation models for gate oxide breakdown," *J. Appl. Phys.*, vol. 86, no. 10, pp. 5757–66, 1999.

[81] S. Cha, W. Kim, and L. Milor, "Gate oxide breakdown parameter extraction with ground and power supply signature measurements," in Proc. Design of Circuits and Integrated Systems (DCIS), 2014.

[82] S. Kim, G. Panagopoulos, C.-H. Ho, M. Katoozi, E. Cannon, and K. Roy, "A compact SPICE model for statistical post-breakdown gate current increase due to TDDB," in Proc. Int. Reli Phys Symp (IRPS), 2013.

[83] G. Wang, K. Cheng, H. Ho, J. Faltermeier, W. Kong, H. Kim, J. Cai, C. Tanner, K. McStay, K. Balasubramanyam, C. Pei, L. Ninomiya, X. Li, K. Winstel, D. Dobuzinsky, M. Naeem, R. Zhang, R. Deschner, M. Brodsky, S. Allen, J. Yates, Y. Feng, P. Marchetti, C. Norris, D. Casarotto, J. Benedict, A. Kniffm, D. Parise, B. Khan, J. Barth, P. Parries, T. Kirihata, J. Norum, and S. Iyer, "A 0.127 μm^2 High Performance 65nm SOI Based embedded DRAM for on-Processor Applications," in Proc. Int. Elec Dev Meeting (IEDM), 2006.

[84] ITRS report (http://www.itrs.net/reports.html).

[85] M. Helm, W. Kavanaugh, B-K. Liew, C. Petti, A. Stolmeijer, M. Ben-tzur, J. Bornstein, J. Lilygren, W. Ting, P. Trammel, J. Allan, G. Gray, M. Hartranft, S. Radigan, J. K. Shanmugan, and R. Shrivastava, "A low cost, microprocessor compatible, 18.4um^2, 6-T bulk cell technology for high speed SRAMs," in Proc. IEEE VLSI Technology, 1993.

[86] W. Kim, C.-C. Chen, D.-H. Kim, and L. Milor, "Built-In Self-Test Methodology with Statistical Analysis for Electrical Diagnosis of Wearout in a Static Random

Access Memory Array," *IEEE Trans. on Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 7, pp. 2521-2534, 2016.

[87] PrimeTime tool: www.synopsys.com.

[88] LEON3 processor: www.gaisler.com.

[89] Mibench benchmark: http://www.eecs.umich.edu/mibench.

[90] B. S. Haran, A. Kumar, L. Adam, J. Chang, V. Basker, S. Kanakasabapathy, D. Horak, S. Fan, J. Chen, J. Faltermeier, S. Seo, M. Burkhardt, S. Burns, S. Halle, S. Holmes, R. Johnson, E. McLellan, T. M. Levin, Y. Zhu, J. Kuss, A. Ebert, J. Cummings, D. Canaperi, S. Paparao, J. Arnold, T. Sparks, C. S. Koay, T. Kanarsky, S. Schmitz, K. Petrillo, R. H. Kim, J. Demarest, L. F. Edge, H. Jagannathan, M. Smalley, N. Berliner, K. Cheng, D. LaTulipe, C. Koburger, S. Mehta, M. Raymond, M. Colburn, T. Spooner, V. Paruchuri, W. Haensch, D. McHerron, and B. Doris, "22 nm technology compatible fully functional 0.1 μm^2 6T-SRAM cell," in Proc. IEEE Int. Elec. Dev. Meeting (IEDM), 2008.

[91] E. van Setten, G. Schiffelers, E. Psara, D. Oorschot, N. Davydova, J. Finders, L. Depre, and V. Farys, "Imaging performance and challenges of 10nm and 7nm Logic nodes with 0.33 NA EUV," in Proc. SPIE 30th European Mask and Lithography Conference, 2014.

[92] R. Kwasnick, A. E. Papathanasiou, M. Reilly, A. Rashid, B. Zaknoon, and J. Falk, "Determination of CPU use conditions," in Proc. IEEE Relia. Phys. Symp. (IRPS), 2011.

[93] N. Aymerich, S. Ganapathy, A. Rubio, R. Canal, and A. Gonza´lez, "Impact of positive bias temperature instability (PBTI) on 3T1D-DRAM cells," *Integration, the VLSI Journal*, vol. 45, Issue 3, pp. 246–252, June 2012.

[94] S. Baeg, P. Chia, S. Wen, and R. Wong, "DRAM failure cases under hot-carrier injection," in Proc. IEEE International Symposium on the Physical and Failure Analysis of Integrated Circuits (IPFA), 2011.

[95] D.-H. Kim, S. Cha, and L. Milor, "Built-in self-test for bias temperature instability, hot-carrier injection, and gate oxide breakdown in embedded DRAMs," *Microelectronics Reliability*, vol. 55, Issues 9-10, pp. 2113-2118, Aug.-Sept. 2015.

[96] S. Yokogawa, S. Uno, I. Kato, H. Tsuchiya, T. Shimizu, and M. Sakamoto, "Statistics of breakdown field and time-dependent dielectric breakdown in contact-to-poly modules," in Proc. IEEE International Reliability Physics Symposium (IRPS), 2011.

[97] H. Tsuchiya and Y. Shinji, "Electromigration lifetimes and void growth at low cumulative failure probability," *Microelectronics Reliability*, vol. 46, no. 9-11, pp. 1415-1420, 2006.

[98] C. Christiansen, B. Li, J. Gill, R. Filippi, and M. Angyal, "Via-depletion electromigration in copper interconnects," *IEEE Trans. Device and Materials Reliability*, vol. 6, no. 2, pp. 163-168, 2006.

[99] A. Das (2014, May 10) , *Exploring Samsung 2x nm LPDDR3 DRAM*, EE Times-Asia [onilne]. Available: http://archive.eetasia.com/www.eetasia.com/ART_8800697748_499486_TA_0f9c7f01_2.HTM.

[100] R. C. Baumann, "Soft errors in advanced semiconductor devices-part I: the three radiation sources," *IEEE Trans. on Device and Materials Reliability*, vol. 1, no. 1, pp. 17-22, Mar. 2001.

[101] D.-H. Kim and L. Milor, "ECC-Aspirin: An ECC-assisted post-package repair scheme for aging errors in DRAMs," in Proc. IEEE VLSI Test Symposium (VTS), 2016.

[102] M. Rab, A. Bawa, and N. Touba, "Improving memory repair by selective row partitioning," in Proc. the Defect and Fault Tolerance in VLSI and Nanotechnology Systems Symposium (DFT), 2009.

[103] J.-P. Son, J. Kim, W. Ahn, S. Han, B.-S. Moon, C. Park, H.-S. Hwang, S.-J. Jang, J. Choi, Y.-H. Jun, and S.-W. Kim, "A highly reliable multi-cell antifuse scheme using DRAM cell capacitors," in Proc. European Solid-State Circuits Conference (ESSCIRC), 2010, pp. 482–485.

[104] B. Jacob, S. Ng, and D. Wang, *Memory Systems: Cache, DRAM, Disk*, Morgan Kaufmann, 2010.

[105] E. Bachman and S. M. Dobrzyuski, "Multiple error correction," *IBM Tech. Disclosure Bull.* 13, No. 8, pp. 2190, 1971.

[106] R.-F. Huang, J.-F. Li, J.-C. Yeh, and C.-W. Wu, "Raisin: Redundancy analysis algorithm simulation," *IEEE Des. Test Comput.*, vol. 24, no. 4, pp. 386-396, Jul.-Aug. 2007.

[107] D.-H. Kim and L. Milor, "Memory yield and lifetime estimation considering aging errors," in Proc. IEEE International Integrated Reliability Workshop (IRW), 2015.

[108] M. Lee, L.-M. Denq, and C.-W. Wu, "A memory built-in self-repair scheme based on configurable spares," *IEEE Trans. CAD of Integ. Cir. and Sys.*, vol. 30, no. 6, pp. 919-929, June 2011.

[109] C.-T. Huang, C.-F. Wu, J.-F. Li, and C.-W. Wu, "Built-in redundancy analysis for memory yield improvement," *IEEE transactions on reliability*, vol. 52, pp. 386-399, Dec. 2003.

[110] L.-T. Wang, C.-W. Wu, and X. Wen, *Design for Testability: VLSI Test Principles and Archtectures*, Morgan Kaufmann, 2006.

[111] C. Park, H. Chung, Y.-S. Lee, J. Kim, J. Lee, M.-S. Chae, D.-H. Jung, S.-H. Choi, S.-Y. Seo, T.-S. Park, J.-H. Shin, J.-H. Cho, S. Lee, K.-W. Song, K.-H. Kim, J.-B. Lee, C. Kim, and S.-I. Cho, "A 512-Mb DDR3 SDRAM prototype with C_{IO} minimization and self-calibration techniques," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 4, pp. 831-838, Apr. 2006.

[112] S.-H. Kim, W.-O. Lee, J.-H. Kim, S.-S. Lee, S.-Y. Hwang, C.-I. Kim, T.-W. Kwon, B.-S. Han, S.-K. Cho, D.-H. Kim, J.-K. Hong, M.-Y. Lee, S.-W. Yin, H.-G. Kim, J.-H. Ahn, Y.-T. Kim, Y.-H. Koh, and J.-S. Kih, "A low power and highly reliable 400Mbps mobile DDR SDRAM with on-chip distributed ECC," in Proc. IEEE Asian Solid-State Circuits Conference (ASSCC), 2007.

[113] M. Qureshi, D.-H. Kim, S. Khan, P. Nair, and O. Mutlu, "AVATAR: A variable-retention-time (VRT) aware refresh for DRAM systems," in Proc. IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2015.

[114] M. Awasthi, M. Shevgoor, K. Sudan, B. Rajendran, R. Balasubramonian, and V. Srinivasan, "Efficient scrub mechanisms for error-prone emerging memories," in Proc. IEEE International Symposium on High-Performance Computer Architecture (HPCA), 2012.

[115] C. W. Slayman, "Cache and memory error detection, correction, and reduction techniques for terrestrial servers and workstations," *IEEE Transactions on Device and Materials Reliability*, vol. 5, no. 3, pp. 397-404, Sept. 2005.

[116] R. Naseer and J. Draper, "Parallel double error correcting code design to mitigate multi-bit upsets in SRAMs," in Proc. European Solid-State Circuits Conference (ESSCIRC), 2008.

[117] T.-Y. Oh, Y.-S. Sohn, S.-J. Bae, M.-S. Park, J.-H. Lim, Y.-K. Cho, D.-H. Kim, D.-M. Kim, H.-R. Kim, H.-J. Kim, J.-H. Kim, J.-K. Kim, Y.-S. Kim, B.-C. Kim, S.-H. Kwak, J.-H. Lee, J.-Y. Lee, C.-H. Shin, Y. Yang, B.-S. Cho, S.-Y. Bang, H.-J. Yang, Y.-R. Choi, G.-S. Moon, C.-G. Park, S.-W. Hwang, J.-D. Lim, K.-I. Park, J. S. Choi, and Y.-H. Jun, "A 7 Gb/s/pin 1 Gbit GDDR5 SDRAM with 2.5 ns bank to bank active time and no bank group restriction," *J. Solid State Circuits*, vol. 46, no. 1, pp. 107-18, 2011.

[118] G. Groeseneken, R. Degraeve, B. Kaczer, and P. Roussel, "Recent Trends in Reliability Assessment of Advanced CMOS Technologies," in Proc. IEEE Int. Conf. on Microelectronic Test Structures, 2005.

[119] E. Miranda and J. Suñé. "Analytic modeling of leakage current through multiple breakdown paths in SiO_2 films," in Proc. Int. Reliability Physics Symp. (IRPS), 2001.

[120] F. Chen, S. Mittl, M. Shinosky, A. Swift, R. Kontra, B. Anderson, J. Aitken, Y. Wang, E. Kinser, M. Kumar, Y. Wang, T. Kane, K.D. Feng, W.K. Hensen, D. Mocuta, and D.-A. Li, "Investigation of emerging middle-of-line poly gate-to-diffusion contant reliability issues," in Proc. IEEE Relia. Phys. Symp. (IRPS), 2012.

[121] F. Chen, C. Graas, M. Shinosky, C. Griffin, R. Dufrense, R Bolam, C. Christiansen, K. Zhao, S. Narasimha, C. Tian, and C.-L. Lou, "New breakdown data generation and analytics methodology to address BEOL and MOL dielectric TDDB process development and technology qualification challenges," in Proc. IEEE Relia. Phys. Symp. (IRPS), 2014.

[122] T. Cahyadi; F. Chen, H. Jiang, S. Mittl, and E. C. Chua, "A correlation study of MOL electrical test method with its physical analysis," in Proc. IEEE Physical and Failure Analysis of Integrated Circuits, 2013.

[123] D.-H. Kim and L. Milor, "A Methodology of Estimating Memory Lifetime Using System-Level Accelerated Life Test and Error Correcting Codes," in Proc. IEEE VLSI Test Symposium (VTS), 2017.

[124] T. Kauerauf, A. Branka, G. Sorrentino, P. Roussel, S. Demuynck, K. Croes, K. Mercha, J. Bömmels, Z. Tőkei, and G. Groeseneken, "Reliability of MOL local interconnects," in Proc. IEEE International Reliability Physics Symposium (IRPS), 2013.